Modelling and characterization of physically unclonable functions

Zouha Cherif

THÈSE

pour obtenir le grade de docteur délivré par :

Présentée et soutenue publiquement par

Zouha CHERIF

le 08 avril 2014

Modélisation et Caractérisation des Fonctions non Clonables Physiquement

Discipline : Microélectronique
Équipes : Systèmes Embarqués Sécurisés - Laboratoire Hubert Curien
Systèmes Électronique Numériques - TELECOM ParisTech

Directeurs de thèse : **Lilian Bossuet, Maître de Conférences HDR, Université Jean Monnet**
Jean-Luc Danger, Directeur d'études, TELECOM ParisTech

Jury

M. Ian O'CONNOR, Professeur, École Centrale Lyon, FRANCE	Président
Mme.Ingrid VERBAUWHEDE , Professeur, K. U. Leuven, BELGIQUE	Rapporteur
M. Bruno ROUZEYRE, Professeur, Université Montpellier 2, FRANCE	Rapporteur
M. Viktor FISCHER, Professeur, Université Jean Monnet saint Etienne, FRANCE	Examinateur
M. Gilles MACARIO-RAT , Docteur, Ingénieur de recherche, Orange Labs, FRANCE	Examinateur

Doctor of Philosophy
in Microelectronics

at the

Modelling and Characterization of Physically Unclonable Functions

Zouha Cherif

April 8, 2014

Research Groups : Secure Embedded Systems - Hubert Curien Laboratory
Digital Electronic systems - TELECOM ParisTech

PhD Supervisors : **Lilian Bossuet, Associated Professor, University of Lyon, Saint-Etienne**
Jean-Luc Danger, Professor, TELECOM ParisTech

A la mémoire de mon père,

***

A ma mère, mes frères et ma sœur,

***

A mon mari et mon fils,

***

A toute ma famille...

Résumé

Les fonctions non clonables physiquement, appelées PUF (Physically Unclonable Functions), représentent une technologie innovante qui permet de résoudre certains problèmes de sécurité et d'identification. Comme pour les empreintes humaines, les PUF permettent de différencier des circuits électroniques car chaque exemplaire produit une signature unique. Ces fonctions peuvent être utilisées pour des applications telles que l'authentification et la génération de clés cryptographiques. La propriété principale que l'on cherche à obtenir avec les PUF est la génération d'une réponse unique qui varie de façon aléatoire d'un circuit à un autre, sans la possibilité de la prédire. Une autre propriété de ces PUF est de toujours reproduire, quelque soit la variation de l'environnement de test, la même réponse à un même défi d'entrée. En plus, une fonction PUF doit être sécurisée contre les attaques qui permettraient de révéler sa réponse. Dans cette thèse, nous nous intéressons aux PUF en silicium profitant des variations inhérentes aux technologies de fabrication des circuits intégrés CMOS. Nous présentons les principales architectures de PUF, leurs propriétés, et les techniques mises en œuvre pour les utiliser dans des applications de sécurité.

Nous présentons d'abord deux nouvelles structures de PUF. La première structure appelée "Loop PUF" est basée sur des chaînes d'éléments à retard contrôlés. Elle consiste à comparer les délais de chaînes à retard identiques qui sont mises en série. Les points forts de cette structure sont la facilité de sa mise en œuvre sur les deux plates-formes ASIC et FPGA, la grande flexibilité pour l'authentification des circuits intégrés ainsi que la génération de clés de chiffrement. La deuxième structure proposée "TERO PUF" est basée sur le principe de cellules temporairement oscillantes. Elle exploite la métastabilité oscillatoire d'éléments couplés en croix, et peut aussi être utilisée pour un générateur vrai d'aléas (TRNG). Plus précisément, la réponse du PUF profite de la métastabilité oscillatoire introduite par une bascule SR lorsque les deux entrées S et R sont connectées au même signal d'entrée. Les résultats expérimentaux montrent le niveau de performances élevé des deux structures de PUF proposées.

Ensuite, afin de comparer équitablement la qualité des différentes PUF à retard, nous proposons une méthode de caractérisation spécifique. Elle est basée sur des mesures statistiques des éléments à retard. Le principal avantage de cette méthode vient de sa capacité à permettre au concepteur d'être sûr que la fonction

PUF aura les performances attendues avant sa mise en œuvre et sa fabrication.

Enfin, en se basant sur les propriétés de non clonabilité et de l'imprévisibilité des PUF, nous présentons de nouvelles techniques d'authentification et de génération de clés de chiffrement en utilisant la "loop PUF" proposée. Les résultats théoriques et expérimentaux montrent l'efficacité des techniques introduites en termes de complexité et de fiabilité.

Abstract

Physically Unclonable Functions, or PUFs, are innovative technologies devoted to solve some security and identification issues. Similarly to a human fingerprint, PUFs allows to identify uniquely electronic devices as they produce an instance-specific signature. Applications as authentication or key generation can take advantage of this embedded function. The main property that we try to obtain from a PUF is the generation of a unique response that varies randomly from one physical device to another without allowing its prediction. Another important property of these PUF is to always reproduce the same response for the same input challenge even in a changing environment. Moreover, the PUF system should be secure against attacks that could reveal its response. In this thesis, we are interested in silicon PUF which take advantage of inherent process variations during the manufacturing of CMOS integrated circuits. We present several PUF constructions, discuss their properties and the implementation techniques to use them in security applications.

We first present two novel PUF structures. The first one, called "Loop PUF" is a delay based PUF which relies on the comparison of delay measurements of identical serial delay chains. The major contribution brought by the use of this structure is its implementation simplicity on both ASIC and FPGA platforms, and its flexibility as it can be used for reliable authentication or key generation. The second proposed structure is a ring-oscillator based PUF cells "TERO PUF". It exploits the oscillatory metastability of cross-coupled elements, and can also be used as True Random Number Generator (TRNG). More precisely, the PUF response takes advantage from the introduced oscillatory metastability of an SR flip-flop when the S and R inputs are connected to the same input signal. Experimental results show the high performance of these two proposed PUF structures.

Second, in order to fairly compare the quality of different delay based PUFs, we propose a specific characterization method. It is based on statistical measurements on basic delay elements. The main benefit of this method is that it allows the designer to be sure that the PUF will meet the expected performances before its implementation and fabrication.

Finally, Based on the unclonability and unpredictability properties of the PUFs, we present new techniques to perform "loop PUF" authentication and

cryptographic key generation. Theoretical and experimental results show the efficiency of the introduced techniques in terms of complexity and reliability.

Remerciement

Ce travail est le fruit de nombreuses rencontres des personnes que je souhaite remercier ici pour toute l'aide, le soutien, les conseils qu'elles m'ont apportés ou tout simplement pour leur bonne humeur et leur joie de vivre qui ont fait de ces trois années de thèse, une aventure exceptionnelle.

J'adresse tout d'abord mes vifs remerciements à **Lilian Bossuet** et **Jean-Luc Danger** mes directeurs de thèse. Merci à Lilian, sa disponibilité, ses qualités humaines et sa bonté resteront pour moi un exemple à suivre. Que ce travail soit le témoignage de mon respect. Merci Jean-Luc de m'avoir accueilli dans son groupe de recherche. Je le remercie d'avoir dirigé cette thèse et de m'avoir soutenu durant ces trois années. Je le remercie enfin autant pour ses qualités pédagogiques et son implication dans ce travail que pour ses qualités humaines, qui ont rendu chaque moment que j'ai passé au laboratoire plus sympathique.

J'espère sincèrement que la collaboration que nous avons eue tous les trois se poursuivra dans le futur.

Je suis reconnaissante à **M. Ian O'connor** pour l'honneur qu'il m'a fait de présider le jury de ce travail.

J'ai eu la joie et la fierté de bénéficier de la participation de **Mme Ingrid Verbauwhede**. Cela a été un grand honneur qu'elle ait accepté de juger mon travail, j'ai admiré sa gentillesse et sa chaleur. Que cette réalisation soit le témoin de ma profonde reconnaissance.

La présence de **M. Bruno Rouzeyre** m'a été précieuse, il m'a impressionné par la précision de ses remarques et de son accueil chaleureux. Je lui exprime mes remerciements les plus sincères en témoignage de ma considération.

Mes remerciements vont aussi à **M. Gilles Macariot-rat** et **M. Viktor Fisher**, de m'avoir honorée d'assister à ma soutenance de thèse et de juger mon travail.

Au cours de toutes ces années j'ai partagé des moments agréables avec des amis exceptionnels : Shivam Bhasin, Houssem Maghrebi, Florent Lozac'h, Youssef Souissi, Molka Ben Romdhane, Tarik Graba, Annelie Husser, Mariem Slimani, Emna Amouri, Nicolas Debande, Jeremie Brunel, Arwa Ben Dhia, Asma Mejri, Salma Belhaj, Laurent Sauvage, Thuy Ngo, Tania, Zakaria Najm, Nidhal Selmane, Pablo, Lubos, Patrick Haddad, Nathalie Bochard, Sebastien Thomas, Taufik Chouta, Olivier Meynard, Aziz Elaabid, Amel Grira,...

Cela était une chance d'avoir une telle bonne compagnie. Ils ont toujours été présents pour partager ensemble le meilleur et le pire.

Ces remerciements ne seraient complets sans mentionner les personnes que j'aime le plus au monde, mon père Mohamed ali, que son Âme repose en paix, tu resteras à jamais gravé dans mon cœur ; ma maman chérie Fethia, je t'admire pour ton soutien continu et ton courage ; mon adorable fils Aziz et mon mari Oualid, je vous adore ; ma bien aimable petite sœur Safa, mes deux frères Wassim et Ahmed et mes adorables neveux Youssef et Brahim, je vous aime ; sans oublier, Mima, Fafa, Mo, Dorra, tata Afifa, am Nourdine, Sabrine, Olfa...

Table des matières

Table des figures

Liste des tableaux

Glossary

AES	Advanced Encryption Standard
ASIC	Application Specific Integrated Circuit
BER	Bit Error Rate
CMOS	Complementary Metal Oxide Semi-conductor
CRP	Challenge Response Pair
DPL	Delay Programmable Line
DUT	Device Under Tests
EMA	Electro-Magnetic Analysis
FF-PUF	Feed-Forward PUF
FIB	Focused Ion Beam
FPGA	Field Gate Programmable Array
HD	Hamming Distance
HW	Hamming Weight
IC	Integrated Circuit
IP	Intellectual Property
KER	Key Error Rate
ML	Machine Learning
NIST	National Institute of Standards and Technology
NSA	National Security Agency
PDF	Probability Distribution Function
PEA	Photonic Emission Analysis
POK	Physically Obfuscated Keys
PUF	Physically Unclonable Function
RAM	Read-Access Memory
RO PUF	Ring-Oscillator PUF
ROM	Read-Only Memory
RSA	Rivest Shamir and Adleman
SCA	Side Channel Analysis
TRNG	True Random Number Generator
TSMC	Taiwan Semiconductor Manufacturing Company
XOR	eXclusive OR

General Introduction

Context and Motivations

Due to the increasing use of the electronic devices in every aspect of day-to-day life and for a wide range of applications, the need for information security has risen exponentially during the last couple of decades. Additionally to the security problems affecting the electronic industry such as intellectual property theft, software piracy and the counterfeit of hardware, the security of electronic devices has become a priority for industrials.
The cryptography is the traditional security technique used to remedy against these security problems. As a function of the system to secure and the nature of the secret information, one or all of the following security measures that cryptography can provide, are applied : authentication, integrity, confidentiality and non-repudiation. However, their security level is highly dependent of the used key in case of encryption, and the identifier in case of authentication.
In order to steal private data or to break an authentication protocol, one type of attack is to retrieve the key/identifier. Therefore, both the generation and the storage of the key/identifier have to be secure against invasive and non-invasive attacks. When the cryptographic key/identifier is stored in a Non-Volatile memory, this opens the door to potential attacks to retrieve the key, including fault attacks to force the access, reverse engineering or probing as proposed by Samyde et al. [SSAQ02]. Moreover, to generate a key some mathematical techniques are used (e.g. PRNG). However they are deterministic and then vulnerable to attacks based on observation (Lenstra et al. [LHA+12]).
To remedy to these security flaws, minimal requirements for a secure key/identifier generation and storage have to be considered :

- Use a source of true randomness that ensures unpredictable and unique keys/identifier.
- Protect the memory from unauthorized parties for a reliable storage of the key/identifier.

The silicon Physically Unclonable Functions (PUFs) seem to be an alternative solution to the traditional cryptographic techniques. They are the main subject of this thesis. We note 3 principle properties that a PUF system has to meet :

- Unpredictability : the generated PUF response varies randomly from one chip to another. But it is static on the same chip.
- Unclonability : the random manufacturing process variation makes the PUF structure very hard to clone.
- Tamper resistance : The PUF has to be robust against physical attacks. For instance invasive attacks should not be able to force the PUF response, or should be detected.

Based on the manufacturing process variations on the CMOS technology, silicon PUFs can provide an on-chip physical functions that yield a device specific identifier/key which is unpredictable and unclonable. Moreover, the PUF response is

not stored. It is generated on demand, thus avoiding the key storage problem.
Our objectives in this research work are to design and characterize silicon PUFs that meet these three properties, and also to make them easy to implement, portable and secure against physical and/or mathematical attacks.
In 2000, Lofstrom et al. [LDT00] propose the first PUF proposal. Since this introduction, at least a new PUF structure is proposed every year. Unlike the NIST [NIS12], the BSI [KS11] or FIPS [FIP01] statistical tests used to evaluate the robustness of TRNG structures, no standard tests has been defined yet to evaluate and compare the PUF performances.
Therefore, in this thesis work, we are also interested in PUF evaluation methods and metrics.

Thesis Outline and Contributions

In this thesis, the focus is laid upon the study of silicon PUF constructions, properties and applications. More concretely, the main contributions are to :

- Devise new PUF structures which are easy to implement and resistant against physical attacks.
- Propose new evaluation method for delay PUFs.
- Explain how PUFs can be used for the generation of cryptographic keys and the authentication of integrated circuits.

This thesis is organized as follows :

Chapter 1 provides a general background about the physically Unclonable Functions (PUFs). It clarifies the PUFs concept and then details the most known properties that a PUF should meet. This starting chapter presents some already proposed applications of a PUF and extensively defines the different PUF classifications proposed in the literature. A deep exploration of the most known intrinsic silicon PUF structures with their own implementation details and performance results is provided. Moreover, an overview of the existing evaluation methods is presented.

Chapter 2 deals with the architectures of silicon delay PUFs. We present a novel structure of delay PUF referred to as “loop PUF”. It is based on identical controllable delay elements. They are serially connected and closed by an inverter to form a single ring oscillator. This chapter presents the implementation strategies and the evaluation performances of both the loop and the arbiter PUFs. We study the effect of the platforms on the performance of delay PUFs when designed on the CMOS 65nm technology. We also present the performance evaluations of the two PUF structures when designed into ASIC and FPGA platforms. The performance analysis is indeed performed under different environmental conditions.

Chapter 3 focuses on the presentation of another novel PUF structure, its implementation on FPGA platforms and the evaluation of its performance. The novel PUF structure is referred to as “TERO PUF”. It takes advantage from the

introduced oscillatory metastability of an SR flip-flop. This chapter details the implementation step required for the validation of the TERO PUF structure. It presents the performance evaluation of the proposed structure when designed on ALTERA Cyclone-II FPGAs.

Chapter 4 proposes a delay PUF performance evaluation method. It uses statistical measurements on delay elements. Its benefit comes from its ability to allow the designer to be sure that her PUF has good performance before its implementation. In this chapter, we provide details about our method by presenting new metrics and evaluation results of both arbiter and loop PUF structures when designed on different platforms.

The topic addressed in Chapter 5 is related to the applications of the proposed loop PUF. In this chapter, we start by presenting our motivations to use the loop PUF for device authentication and cryptographic key generation purposes. Then, we detail the proposed device authentication procedure using the loop PUF and show the obtained results when tested on ASIC platforms. The proposed method is based on the measurement of physical values of delay elements. These physical values are used to authenticate devices since they are much more precise than the binary response of the loop PUF. The proposed authentication method is indeed based on the Pearson correlation coefficient. It takes into account both offset and scaling phenomenon that can affect the PUF response. Finally, we describe the key generation procedure which has been developed for the loop PUF. Then, we show and discuss some of the obtained results when tested on different PUFs on ASICs. The proposed method can be divided into two stages : the profiling and the key generation steps. In order to enhance the reliability of the generated key, we propose to use a dynamic reliability analysis procedure. However, this is not sufficient to guaranty the regeneration of the reference key. Therefore, we propose a key correction procedure that is based on both Hamming codes and Chase algorithm.

The last chapter provides general concluding remarks and highlights some perspectives for future research.

Chapitre 1

Physically Unclonable Functions : Basics

In this chapter, we focus on clarifying the concept of Physically Unclonable Functions (PUFs). This is important due to the increasing number of the proposed PUFs structures. Moreover, there are functions that meet the PUFs properties without being called PUFs. Some of them were presented before the emergence of the PUF concept. In what follows, we first discuss the PUF concept. Second, we detail the most important properties that a PUF has to meet, and also discuss different possible usages in different applications. Then, we extensively present the different PUFs classifications proposed in the literature. We thereafter provide a deep exploration of most known intrinsic PUF structures with their implementation details and a performance analysis. And finally, an overview of existing evaluation methods is also presented.

1.1 PUFs : Concept, Properties and Applications

1.1.1 Concept of PUFs

The PUF concept is first introduced by Lofstrom et al. [LDT00] in 2000. The authors propose to exploit the mismatch in silicon devices for integrated circuit (IC) identification. In 2001, Pappu [Pap01] introduced them as Physical On-Way Functions. Then, Gassend et al. [GCvDD02] propose a Silicon Physical Random Functions and present it as a PUF. The reason for this choice of name is to avoid the acronym "PRF" and therefore confusion with "Pseudo Random Functions". A PUF is a function (not mathematical) embedded in a physical device in order to extract a secret from a complex physical system. We can describe it as a function which returns a characteristic value (as a signature or a fingerprint or a DNA, *response*) of an integrated circuit for a given *challenge*. Hence, we can define it as a *Physical challenge-response procedure* to extract the signature of an integrated circuit. A PUF is similar to the human fingerprints since it produces a specific device signature to allow the device to be authenticated. Also, it mainly takes advantage from inherent CMOS variations, as this kind of DNA is inherent to the device and cannot be cloned.

1.1.2 Properties and Parameters of PUFs

When studying the first definition of PUFs as given by Gassend et al. [GCvDD02], we can extract two main properties that a PUF should meet. The first property is **unpredictability**. A PUF is recognized to be unpredictable when an attacker who can use a limited and fixed amount of resources can only extract a negligible amount of information from the PUF's secret response. The second property is **physical unclonability**, also called **manufacturer resistant**. This means that, it should be technically very hard, not to say impossible, to produce two identical PUFs. PUFs take advantage from the circuit characteristics, which are related to the uncontrollable random variation on the manufacturing process. Therefore, the less is the control of the circuit during the manufacturing process, the harder is the reproduction of an identical PUF.

Later on, Maes et al. [Mae12] extend the property list that a PUF should fulfill. They present four other properties which have been identified from multiple proposed PUF definitions given in the literature. These properties are described below :

1. **Evaluable** or low cost. This means that from a practical point of view, the used measurement circuit should be easy to implement and very low cost (e.g using standard components). From a theoretical point of view, the PUF response should be easy to evaluate/produce (e.g. evaluable using fixed and limited amount of time).

2. **Unique**. The PUF response is extracted from the identity of the physical entity. Then, in theory, a set of challenge-response pair should be sufficient to uniquely identify a PUF among a given population.

3. **Reproducible** or steady. This property distinguishes PUFs from True Random Number Generators (TRNGs). In fact, the PUF response should be reproducible, up to small error, when introducing the same challenge even when are re-asked under different environmental conditions.

4. **Secure**. This property can be divided into three properties :
   - Mathematically unclonable. This means than, given a PUF, it is hard to construct a software function able to reproduce all challenge-response pairs up to small errors.
   - One-way. Given the PUF response, it is hard to predict or to compute the applied challenge.
   - Tamper-evident. To avoid cloning PUFs when an invasive physical attack is performed, PUFs should produce error response when asked.

Recently, Handshuh et al. [HST10] proposed another interesting property called "**randomness**". It evaluates the statistical quality of the generated secret response. In order to extract a high-quality secret key from a PUF, a sufficient amount of randomness is needed in the PUF responses.

Note that among all proposed PUFs in the literature, No PUF verifies all these properties at 100%. Most of them meet nearly all of the needed properties up to a certain percentage. However, we assume that both, the uniqueness and the reproducibility (or steadiness) properties are the most important to verify. Therefore, we distinguish two important parameters, the intra-distance and the inter-distance variations, to evaluate the uniqueness and the steadiness of a PUF structure, respectively. Ideally, the intra-distance parameter should be studied under both normal and variable environmental conditions.

1.1.2.1 Intra-distance or Steadiness Evaluation

The Intra-distance is a random value which describes the distance between responses of the **same PUF** when applying the **same challenge**. Figure 1.1 illustrates the steadiness evaluation process. To better characterize the steadiness of a PUF instance, it is very important to know the distribution of this random value on both normal and variable environmental conditions. Thus, statistics of this distribution are often used as a metric to evaluate the reproducibility of a PUF instance. Note that, the mean value of the intra-distance variation is higher when evaluated under variable conditions. Therefore, to better evaluate the PUFs steadiness, it is appropriate to take into account the worst-case. The latter is represented by the largest intra-distance.

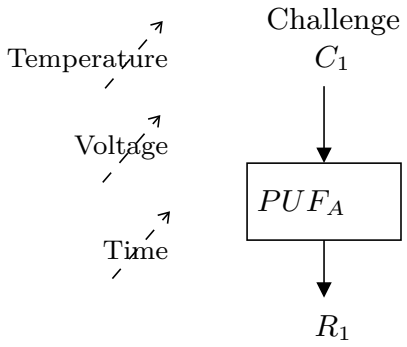
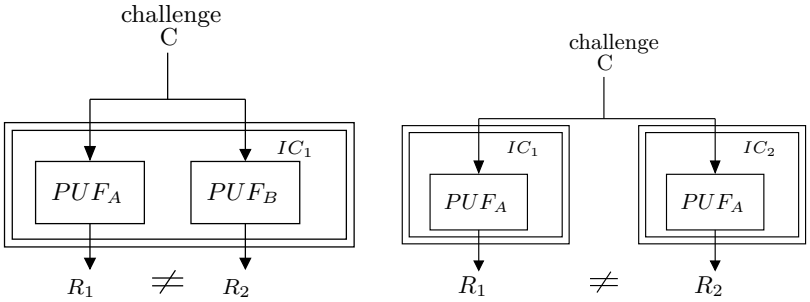


FIGURE 1.1 – Steadiness evaluation process.

1.1.2.2 Inter-distance or Uniqueness Evaluation

The Inter-distance is also a random value which describes the distance between responses of **different PUFs** (included in the same device or on different devices) when applying the **same challenge**. Figures 1.2a and 1.2b illustrate the uniqueness evaluation process. In this case, to better characterize the uniqueness of a PUF instance, it is very important to know the distribution of this random value on normal environmental conditions. Therefore, statistics of this distribution are often used as a metric to evaluate the uniqueness of a PUF instance.



(a) Intra-uniqueness evaluation. (b) Inter-uniqueness evaluation.

FIGURE 1.2 – Uniqueness evaluation process.

1.1.3 PUF Applications

Based on the existing literature, we distinguish four principle application scenarios that can use a PUF instance.

1.1.3.1 Document Identification and Device Authentication

For this application scenario, the PUFs are used essentially to fight against counterfeiting of documents and devices. Hereafter, we explain how this can be

done using PUFs.

Document Identification
Before the emergence of the PUF concept, Simmons [Sim91] proposed a technique to identify the documents using the internal fingerprint of the paper-based object (e.g. bills). Two essential steps are needed to generate the paper identifier.

- Extract the physical fingerprint of the paper. The fingerprint depends on random arrangement of fibers of random lengths and orientations in the paper. Since the latter is produced with random lengths of optical fibers in the pulp.
- Generate the secret information about the paper by applying a data compression technique.

The information obtained from the paper fingerprint is quite low, since there is a redundancy. Each fiber produces two correlated outputs as each end is illuminated. Therefore the author proposes to add an additional step to uniquely identify the paper.

Low-cost Device Authentication
Suh et al. [SD07] proposed a low-cost device authentication based on a challenge-response protocol. The described method can be applied even to resource constrained platforms such as RFIDs. This authentication mechanism ensures that an adversary cannot obtain the PUF output used for authentication. In fact, the exploited PUFs are expected to have exponential numbers and non-linear challenge-response pairs. Then, only using a challenge once, we are able to avoid men-in-the-middle-attacks. Also, based on the non linearity of the PUF entity, we make PUFs model-building harder. To run the proposed authentication process, the following three steps are needed :

1. First, the trusted party (manufacturer) applies a randomly chosen challenge to get the corresponding response.
2. Then, the trusted party stores the challenge-response pairs in a database for an eventual future authentication process.
3. Finally, to check the authenticity of a device after recovering it from the hardware market, the trusted party selects a challenge and then obtains the corresponding PUF response. The condition on the challenge is that it has been recorded but never been used for authentication purpose. Otherwise, the recovered IC is a fake one. If the response matches with the previously recorded one, the IC is authentic.

1.1.3.2 Cryptographic Key Generation

Some cryptographic primitives require a key that satisfies specific mathematical properties such as the RSA algorithm. Dodis et al. [DRS04] and [DORS08]

present the secure sketch principle that allows a reliable key extraction which is inspired by biometric methods. This method has also been exploited by Suh et al. [SD07] who propose a PUF post-process to generate a reliable and specific cryptographic key. The cryptographic key generation process is divided into two stages. The first stage is the initialization, or enrolment, and the second stage is the regeneration of the PUF response. Figure 1.3 shows the overall cryptographic key generation process using PUFs. The generated cryptographic key can be used with Advanced Encryption Standard (AES), Rivest Shamir and Adleman (RSA) or other cryptographic algorithms. Intrinsic-ID [II] propose to use such generated key to secure the personal data in the cloud using the AES cryptographic algorithm.

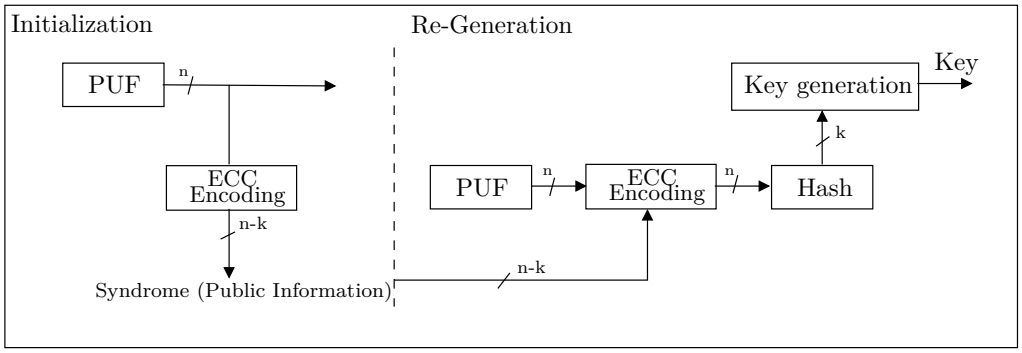


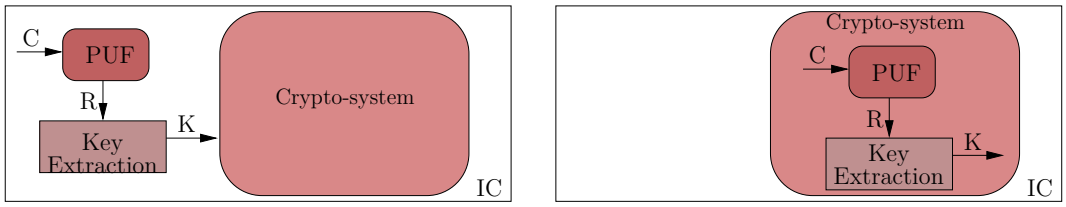
FIGURE 1.3 – Cryptographic key generation using PUFs.

1.1.3.3 Hardware Entangled Cryptography

Maes et al. [MV10] propose to use PUFs to generate a cryptographic key within an existing cryptographic primitive. In this case, the cryptographic primitive fully integrates the PUF. Hence, the hardware entangled cryptographic primitives are keyless. The secret key is not stored in memory, neither in non-volatile memory or volatile memory. Therefore, we can avoid not only non-volatile memory attacks but also volatile memory ones. Figures 1.4a and 1.4b show the difference between the classical cryptography key generation process using the PUF and the hardware entangled cryptography.

1.1.3.4 Intellectual Property (IP) Protection

Based on the Physically Obfuscated Keys "POKs" principle, PUFs can be also used to protect programmable hardware IP blocks against piracy. POKs were first introduced by Gassend et al. [Gas03]. The idea is to generate a device specific key (stored in a physical way) that can be used to decrypt the same algorithm (stored in ROM) in different devices. This ensures that such an algorithm cannot



(a) Classical cryptographic key. (b) Hardware entangled cryptography

FIGURE 1.4 – Classical cryptographic key generation vs. hardware entangled cryptography using PUFs.

turn correct in another cloned device. In fact, first, the ROM is encrypted using a same key K in different devices. Second, for decryption, to generate the same key K a hardwired PUF with a challenge is used together with the contents of some fuses (Figure 1.5). Since the response of the PUF is different from a chip to another, by setting the appropriate bit on each fuse, the decoding key K is generated to decode the ROM. We note that an attacker can read the fuses state. But, even when knowing the state of the fuses, the value of K will remain secret and illegal copies of the chip will not work. However, the secret information is present on the chip. It can be thus cloned when performing an invasive attack.

Bringer et al. [BCI09] generalizes the POK concept. They propose to combine both masking techniques and POKs to increase chips resistance against physical threats.

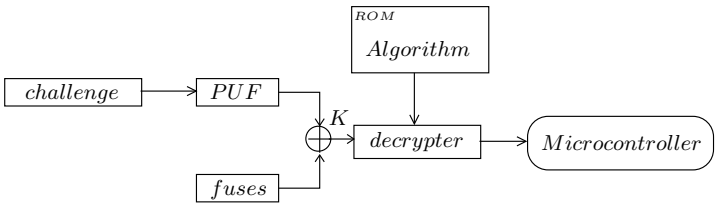


FIGURE 1.5 – IP protection using PUF.

1.2 PUFs Classification

In the literature, PUFs are classified differently in depending on the considered criteria. We distinguish three possible classifications ; each one contains two main types.

- First, they can be classified depending on their construction **material**, based on electronic or non-electronic material.
- Second, considering the **properties** when constructed, we can identify two types : Intrinsic and non-intrinsic PUFs.

– Finally, we can divide PUFs into two types, weak and strong PUFs, by evaluating the **security** level of the challenge-response behavior.

1.2.1 Non-electronic and Electronic PUFs

1.2.1.1 Non-electronic PUFs

This category contains all PUFs which are based on non-electronic material or technology. Note that these types are the origin of the PUF behavior. We found those based on the random fiber-structure of papers (paper PUFs by Simmons [Sim91]) and optical-based PUFs which are based on the random reflection of beams (optical PUFs by Pappu et al. [PRTG02] see Figure 1.6).

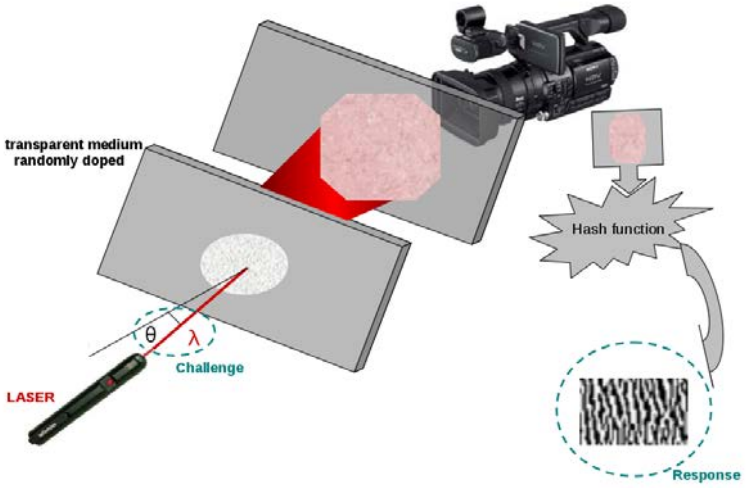


FIGURE 1.6 – Optical PUF principle.

1.2.1.2 Electronic PUFs

Electronic PUFs take advantage from the random variation of electronic materials to generate the PUF response. In this category we can find radio-frequency based PUFs (DeJean et al. [DK07], Guajardo et al. [GvT+09]) and silicon PUFs. Guajardo et al. [GvT+09] propose the LC-PUF. When an electromagnetic radio frequency field is generated around the antenna, the circuit absorbs an amount of power that depends on the frequency and on the characteristics of the circuit capacities.

Silicon PUFs (Section 1.4) are the major subclass of electronic PUFs. It includes electronic circuit PUFs embedded on a silicon chip. Since silicon PUFs can be easily connected to another embedded system on the same chip, they are widely used for security solutions and they are the main type of PUFs. We focus on this type in this thesis.

1.2.2 Non-intrinsic and Intrinsic PUFs

Guajardo et al. [GKJST07] propose a novel classification of PUFs depending on their construction properties. They introduce the notion of *intrinsic* PUFs. The authors define an intrinsic PUF as "a PUF generating circuits already present in the device and that requires no modification to satisfy the security goals". Later on, Maes [Mae12] extends this notion to consider PUFs as intrinsic only when they meet at least the following two conditions :

1. The required measurements to generate the PUF response should be performed "*internally*" by embedded measurement equipment.
2. They have to take advantage from only implicitly introduced randomness during the manufacturing process called process variation.

1.2.2.1 Internal and External Measurements

To extract the PUF response, the PUF designer should use measurement equipment. We distinguish two ways to measure the PUF response :

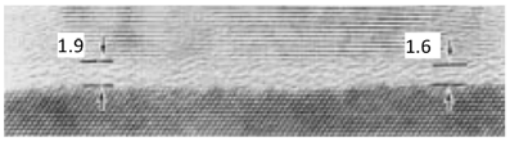
- **External** measurements are performed using equipment external to the PUF entity.
- **Internal** measurements are performed using equipment embedded to the PUF entity.

The internal measurements provide two main advantages. First, they are more precise since measurement equipment are embedded in the PUF entity. Then, there is less influence by the outside noise. Second, since measurements are built internally, the response can be considered as an internal secret which cannot be revealed as long as it is not released. Then, we can deduce that with the internal measurements, the PUF structure is more secure. However, the user cannot verify and control the measurements when they are performed.

1.2.2.2 Implicitly and Explicitly Introduced Randomness

The basic principle of PUFs is to take advantage from the randomness introduced into the entity. We distinguish two kinds of randomness used by PUFs. The first kind is the randomness added explicitly to the PUF entity. Since it requires an explicitly randomization procedure, it is more costly in time point of view than the implicitly introduced randomness. The latter technique is based on the undesirable randomness introduced explicitly to the circuit in the manufacturing procedure. It is very interesting since it already exists and nothing has to be explicitly added to the circuit during manufacturing. The implicitly introduced randomness is the result of implicit variations during the manufacturing process. It includes shifts in the values of some parameters such as the effective channel length, the oxide thickness, the dopant concentration, the inter-layer dielectric

thickness. Figure 1.7a and 1.7b show the process variation of the gates oxide thickness and the dopant concentration in a transistor, respectively.



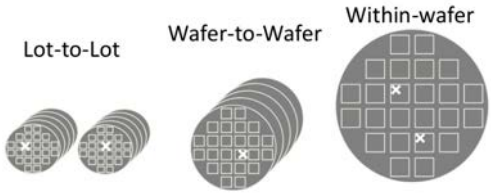
(a) Variation on the gates oxide thickness.

(b) Variation on the dopant concentration.

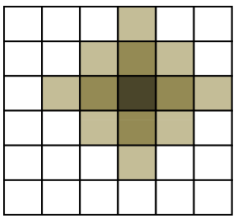
FIGURE 1.7 – Examples of process variation [Sap11].

Sapatnekar [Sap11] classify the process variations phenomenon into two categories depending on their physical range on a die or wafer.

- Die-to-die variations correspond to the changes from one die to another (Figure 1.8a).
- Within-die variations correspond to variability within a single die (Figure 1.8b).



(a) Die-to-die variations.



(b) Within die variation.

FIGURE 1.8 – Die-to-die vs. within die variations [Sap11].

Based in this classification, the optical PUF is considered as a non-intrinsic PUF since it is externally evaluated and its random features are explicitly introduced.

1.2.3 Strong and Weak PUFs

The distinction between strong and weak PUFs is first discussed by Guajardo et al. [GKJST07]. Based on the number of the challenge-response pair, the authors present a novel way to classify PUFs. A PUF is called strong when its

challenge is very large such as the optical PUF, the arbiter PUF, etc. Otherwise, the PUF is called weak PUF. Weak PUF structures have essentially one challenge such as SRAM PUFs. They are mainly used for the generation of cryptographic keys since it uses fixed challenge.

1.3 Attacks on Silicon PUFs

In this work, we are interested on the study of silicon PUFs. In this section we present the possible attacks proposed on the literature in order to clone the PUF or to extract the PUF response either mathematically or physically.

1.3.1 Modeling Attack

The modeling attack on PUFs is a non-invasive attack. When it is applied, by simple simulation, we can predict the response of the PUF and then break its security. Such an attack can be only applied for some of the strong PUFs. It presumes that an attacker is able to :

1. Collect a set (non-negligible) of Challenge-Response Pairs (CRPs) which is not possible for weak PUFs since they have one challenge.
2. Build a numerical model of the PUF.
3. Predict with high probability the PUF response to an arbitrary chosen challenge.

The success rate (hit probability of prediction) of the modeling attack is closely related to the model of the PUF. It is known that PUFs with linear models can be attacked easier than those with non-linear models. However the complexity of the model to build is highly correlated with the complexity of the PUF architecture (e.g. the number of stages, the length of the challenge, etc). Ruhrmair et al. [RSS+10] attest that machine learning (ML) techniques are a powerful tool for such modeling attacks. In Section 1.4, we show the obtained results in the literature of some modeled silicon PUFs. Let us denote by $p_{success\_rate}$ the prediction success rate, and q_{nb_tries} the number of known CRPs used for training. Then, we can say that a PUF is ($p_{success\_rate}$, q_{nb_tries})-modelable.

1.3.2 Side-Channel Analysis

Side-channel analyses (SCA) are a form of physical attacks on devices (Kocher et al. [KJJ99]). A side-channel attack is a passive attack. It does not disturb the system resources and behavior. The idea is to exploit the physical information leaked by the system while it is operating. Every implementation causes indeed additional effects while operating, e.g. power consumption or electromagnetic radiation or photonic radiation. Putting information collected by these

side-channels in correlation with the supposed activity, makes it possible to exploit information about internally used information. This technique is first proposed to attack a hardware implementation of cryptographic ciphers. The goal is to extract the secret key. The same principles are used to attack the PUF structures in order to extract their responses (outputs). In the next section we present the vulnerabilities of some silicon PUFs to this kind of attacks.

1.4 Silicon PUF Structures

We can classify most known silicon PUFs into two main classes based on their operating principles :

- Delay-based silicon PUFs : they take advantage from the random variation on the delay of wires and components on a digital circuit.
- Memory-based PUFs are the second class of silicon PUFs. They use the device mismatch phenomenon in bi-stable memory elements as random variation to generate the device signature.

For each class of silicon PUF, most known types are presented hereafter : arbiter PUF, RO-PUF, SRAM PUFs, latch and butterfly PUFs.

1.4.1 Delay based PUFs

1.4.1.1 Arbiter PUFs

Basic Structure and Extended Implementations

The arbiter PUF is a delay based silicon PUF. It was proposed by Gassend et al. [Lim04], [GCvDD02] and [LLG+04]. The arbiter PUF structure is composed of two parallel, identical and controllable delay paths and an arbiter circuit at the end. Figure 1.9 shows the arbiter PUF structure as proposed by Gassend et al. [Lim04], [GCvDD02] and [LLG+04]. It is composed of a sequence of switch components. The simplest way to implement them is with a pair of 2-to-1 multiplexers. Each one interconnect two input ports to two output ports with different configurations (straight or crossed) depending on the applied control bit (0 or 1).

The idea is to introduce an edge and then make a race between the two paths and sample the top signal by the bottom one at the end. Then, the arbiter circuit outputs a binary value to indicate which one of the two paths is the faster (or the slower). Since the two parallel paths are identical, the delay difference between them is minor. Hence, two scenarios are possible to get the arbiter circuit decision :

1. Even when designed identically, the delays of the two paths may be not equal. This is due to random silicon process variation. This random difference between the high and the low path will determine the output of the

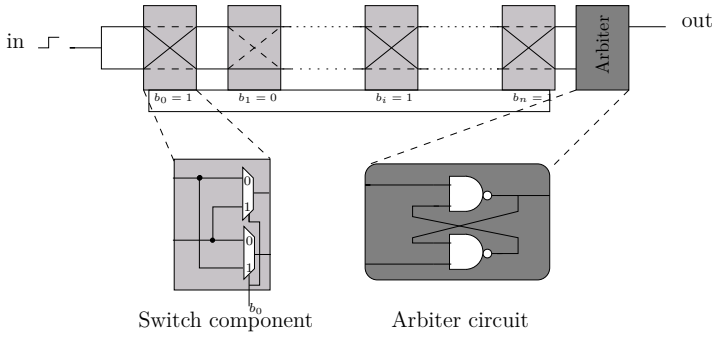


FIGURE 1.9 – Basic arbiter PUF structure.

arbiter circuit and then the PUF response. And, since the random silicon process variation is device specific, the arbiter output will be device specific.

2. It is possible also that even with the random process variation between the two designed paths, the delay difference cannot be detectable by the arbiter circuit. Therefore, the introduced edge will reach simultaneously the two inputs of the arbiter circuit and will make the arbiter in a metastable state. Then, after a short random time the arbiter will outputs a value which is independent from the paths' race.

The proposed architecture of the arbiter PUF is very hard to implement. The two paths have to be identical. Placement and routing constraints are needed. Therefore, Ozturk et al. [OHS08] propose another way to implement the two parallel paths to make it easier to implement in an FPGA platform. The authors propose to use a pair of tristate buffer circuit as a switch component to avoid cross coupled wires (Figure 1.10).

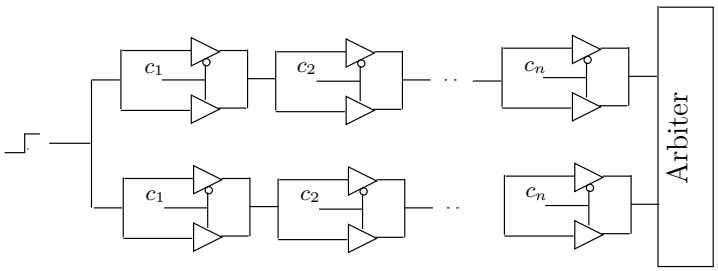


FIGURE 1.10 – Tristate PUF structure.

Another solution is provided by Majzoobi et al. [MKD10] to avoid routing constraints. The authors propose also a new non swapping switch structure using Delay Programmable Lines (DPLs). Majzoobi et al. [MKD10] propose to insert tuning blocks to both up and low paths to cancel out the delay bias caused by the routing asymmetry. The tuning blocks are implemented as the switch blocks

(using DPLs). The difference between them is that the control bit applied to the top path is not the same applied to the bottom path for the tuning blocks (see Figure 1.11). For presentation issue, we replace each basic controllable delay elements by a triangle.

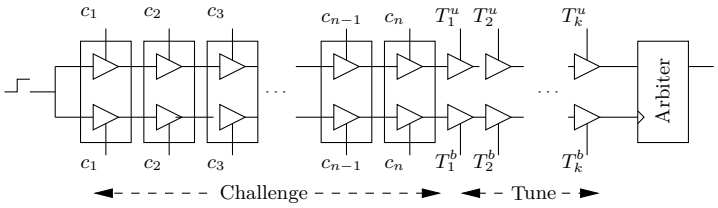


FIGURE 1.11 – Arbiter PUF based on DPLs.

Arbiter PUF Vulnerabilities

If we assume that, for the first proposed structure of the arbiter PUF the two paths are composed of n switch components, then 2^n different possibilities of control bits (called challenges) can be applied. The bit response of the arbiter PUF is linearly dependent from the n delay elements (switch blocks). This means that the 2^n bit-responses cannot be independent. We can model the circuit as an additive model. Considering that the delay of the path is the sum of all elementary delays, and once one learns these elementary delays and the relation with the applied challenge bits, one can be able to predict the response bit to a given random challenge even when we do not have an access to the PUF. Then, we can say that the arbiter PUF structure is mathematically clonable. Also, the arbiter PUFs either based on tristate buffers or DPLs are equivalent to switch based arbiter PUF in terms of security. The challenge-response pairs can be modeled by attackers since they are linearly dependent. Lee et al. [LLG+04] and Ruhrmair et al. [RSS+10] attest that the arbiter PUF basic structure can be modeled using a machine learning technique. To make the arbiter PUF structure much more secure against modeling attacks a few structures have been proposed. First, Gassend et al. [GLC+04] improved the arbiter structure by adding some combinatorial components to introduce a non linearity on the arbiter PUF structure. The novel proposed structure is called feed-forward arbiter PUF referred to as "FF-arbiter PUF" (Figure 1.12). In fact, an arbiter (a latch) is added to the first arbiter PUF structure and it is placed at an intermediate point on the circuit. Its output drives another switch component later in the arbiter PUF main structure. As much as needed, we can add intermediate arbiters.

This makes the arbiter PUF challenge-response pair increasingly non linear. But the problem of this solution is that it decreases the reliability of the PUF since if an error, due to noise, of an intermediate stage was accrued, the noise will increase in the final PUF response. Second, Suh et al. [SD07] propose to obfuscate the output of the arbiter PUF by Xoring the outputs of multiple arbiter PUFs

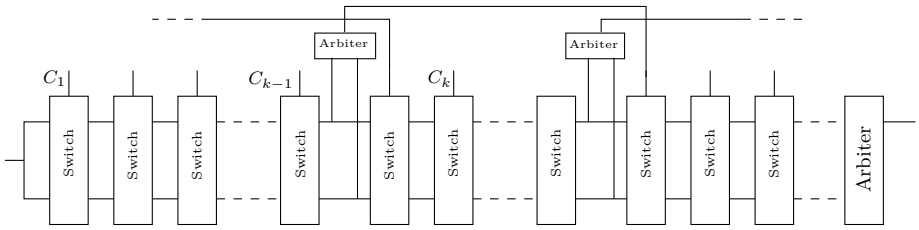


FIGURE 1.12 – FF-arbiter PUF structure.

(Figure 1.13).

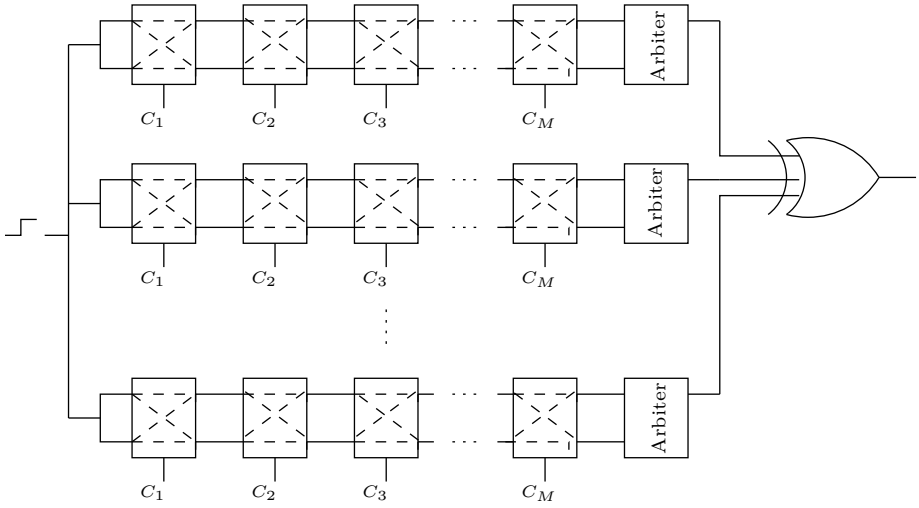


FIGURE 1.13 – Xor-arbiter PUF structure.

Evaluation and Experimental Results

Gassend et al. [GLC+04] implement and test the basic arbiter PUF structure on a set of 23 FPGAS. The evaluation of the inter and the intra-distance are estimated by 1.05% and 0.098%, respectively. The inter-distance is very low. This means that the arbiter PUF implementation is biased. This comes from the difficulty to add routing and placement constraints in an FPGA platform. The same structure, when tested on a set of 37 ASICs, presents better characteristics. It presents an inter-distance of 23% which is still less than the ideal case of 50% and a 0.7% of intra-chip variation under normal environmental conditions.

Later on, the robustness of the same basic implementation against modeling attack have been studied by Lim et al. [Lim04] The authors demonstrate that this structure presents a 3.55% as prediction error rate using 5000 tries. We can therefore say that the PUF is (96.45%, 5000)-modelable. The authors also demonstrate that the FF-arbiter PUF presents an average of 40% of inter-distance

and 2.2% of intra-distance. Based on simulated implementations on FF-arbiter PUF, Ruhrmair et al. [RSS+10] show that we need 50000 tries to attend at least a success rate of 97.43%. It depends on the number of used switch components and the number of FF-arbiters. We can thus say that the FF-arbiter PUF is (97.43%, 50 000)-modelable. As a consequence, the FF-arbiter PUF is more robust than a basic structure of an arbiter PUF. However, it is still attackable by modeling.

1.4.1.2 RO-PUF PUFs

Basic Structures of a RO-PUF

The second most known type of silicon PUF is the ring-oscillators based PUF. It is based on measuring frequencies of digital oscillator circuits. Since there is a random silicon process variation on the delay of digital components, the measured frequencies of oscillating circuits are random. The basic structure of a RO-PUF is mainly composed of a single ring-oscillator and a counter.

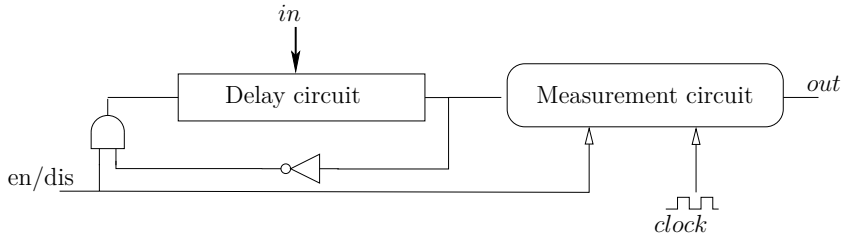


FIGURE 1.14 – Basic RO-PUF structure.

The first structure was proposed by Gassend et al. [GCvDD02]. As shown in Figure 1.14, the proposed RO-PUF is composed of a single ring-oscillator which is based on a single delay circuit closed by an inverter and a frequency measurement circuit. The delay circuit they use is composed of multiple switch components as used for the arbiter PUF (cf. Section 1.4.1.1). Unlike the arbiter PUF structure, there is no need for routing constraints. An external input signal is used to enable/disable frequency measurements by enabling/disabling both the oscillation of the delay circuit and the counter circuit. The authors propose to use a synchronous counter to measure the oscillating frequency. Therefore, a register and an AND-gate are used to detect rising edges. To perform consistent measurements, one constraint must be maintained. The clock signal must be at least twice faster than the delay circuit.

Gassend et al. [GCvDD02] show that when two ring-oscillator PUFs are equally implemented in two different FPGAs, and mainly due to process variation, the measured frequencies are different. However, the experiments realized by the authors show that the influence of the environmental variations is much more important than the process variation. Indeed, when the temperature or the supply voltage changes, delays of similar design vary proportionally to each

others. Therefore, the authors propose to make changes on the PUF structure. They propose, first, to implement two identical loops in the same device, second, to measure simultaneously the oscillation frequencies. Then, the obtained ratios when dividing the two measured frequencies by one another called *compensated measurements* can be considered as an eventual PUF response which is more stable. This solution is the same adopted for the arbiter PUF, since we are interested on the winner of the race between two identical parallel paths (it is a differential measurement).

To make the dependency between the two parallel paths used as a delay circuit more difficult, Gassend et al. [GCvDD02] propose to add delay buffers to the two delay paths. Using the delay buffers, if an elementary delay in the circuit becomes faster, the overall circuit can become slower. Therefore, it is hard for an attacker to model the PUF behavior. However, designing a circuit with the delay buffers is very delicate. If they are misused, the circuit can become chaotic and then vulnerable against modeling attacks. Moreover, the proposed PUF structure presents another minor drawback. The generated PUF response is a real or an integer value depending on whether we apply the *compensated measurements* technique or not. It cannot be used directly as a bit string response. Hence, a quantifier must be used as a post processing operation to generate a bit string PUF response.

Performance Analysis and Extended Implementations of RO-PUFs
Latter on, Suh et al. [SD07] propose another structure of PUF based on ring-oscillators as shown in Figure 1.15. The delay circuit used in the basic ring-oscillator circuit structure is replaced by a loop composed of nb_{inv} inverters (with nb_{inv} odd). This loop is duplicated n times. Due to random process variation, their oscillation frequencies are different. The PUF response is directly correlated to the oscillation frequencies. First, to select a pair of ring-oscillators, a bit challenge is introduced as a bit selection on the two *n-to-*1-mux. Second, the oscillation frequencies of the two selected loops are measured using two counters. Finally, the two measured frequencies are compared to generate one bit which can be considered as the PUF response. Since each comparison of a pair of oscillators generates a bit response, the length of the PUF response can reach $\frac{N(N-1)}{2}$ bits. However, considering the $\frac{N(N-1)}{2}$ distinct pairs, the number of independent bit response are lower than $\frac{N(N-1)}{2}$. For example, if we consider three ring-oscillators A, B and C. If A is faster than B, and B is faster than C, then it is clear that the oscillator A is faster than C. Hence, we conclude that last generated bit is correlated with the previous two. To avoid correlated bit responses with simplicity, Suh et al. [SD07] proposed to use each ring-oscillator only once. In this way, considering n ring-oscillators, we are able to generate $n/2$ independent bits. Unfortunately, even when the PUF response depends on the relative comparison between two implemented ring-oscillators, errors can occur due to environmen-

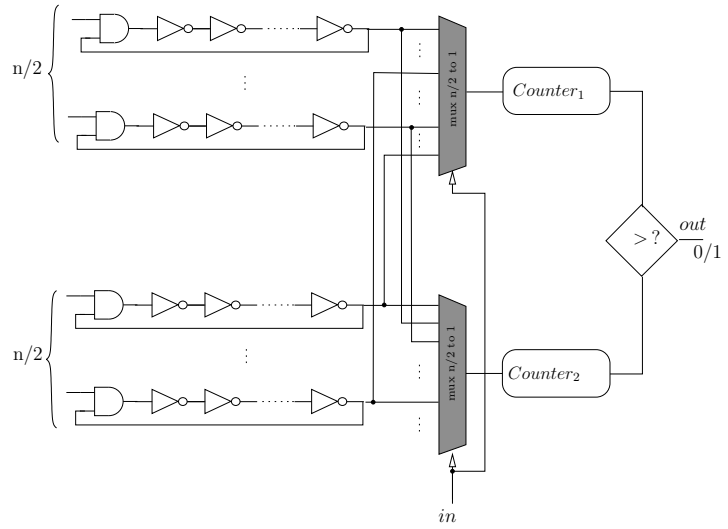


FIGURE 1.15 – An improvement of the RO-PUF structure.

tal variations. Then, to reduce the error rate of generated bit-responses, authors propose to select pairs of oscillators in such way that the selected pairs has a maximum distance. The applied technique is called 1-*out-of-k masking*. In fact, they propose to evaluate k oscillators and then select only the pair with the largest distance and consider the output of this comparison as the bit PUF response. However, this solution reduces more the RO-PUF response length. Considering n oscillators only n/k bit-responses are generated. To evaluate the reliability of the proposed structure, experiments have been carried out on 15 FPGAs with the same model of 1024 ring-oscillators. With $k = 8$, the technique 1-*out-of*-8 *masking* is used. Experiments show that the proposed PUF structure produces a unique response. When the same challenge is applied for two FPGAs, they produce two different responses. A percentage of inter-distance of 46.15% is obtained for the RO-PUF which is very close to the ideal percentage of 50% On the other hand, an intra-distance of 0.48% is obtained which is not far from the ideal percentage of 0% even in worst environmental conditions ($Temp = 120°C$, $V = V_{dd} + 10\%$).

Maiti et al. [MCMS10] present large scale characterization results of the RO-PUFs design presented by Suh et al. [SD07]. They also made the measurements dataset publicly available in [Tec]. Experiments have been carried out a large population of FPGAs (125 FPGAs). We note that the 1-*out-of-k* masking technique is not applied. Then, $n-1$ bit-responses are generated from n implemented ring-oscillators. Authors conclude that, at normal environmental conditions, the studied structure presents an average inter-distance of 47.31% and an average intra-distance of 0.86%. However, the intra-distance results when changing specially voltage conditions (reduced by 20%) goes up to 15% which represent a low

reliability percentage. With the temperature variation ($25°C$ up to $65°C$), the intra-die variation remains almost stable.

Later on, Maiti et al. [MS09] [MS11] propose a configurable ring-oscillator design. A bit-string challenge allows the user to select one path among multiple ones. In fact, one CLB (1 CLB=4 Slices in a Xilinx SpartanII FPGAs) is used to design a single ring-oscillator. Two inverters and a multiplexer are implemented inside a single FPGA Slice. Then, a bit-string challenge allows the designer to select which inverters will be used inside each Slice. Moreover, they propose a very efficient but costly effective technique to reduce the effect of environmental variations. The idea is to identically configure ring-oscillators implemented into two different CLBs by applying the same control inputs (challenge). Thus, considering two identically controlled oscillators as a pair, they propose to select the ring-oscillator pair which has the maximum frequency difference. This achieves better reliability results than the original masking technique applied by Suh et al. [SD07] without any post-processing operations.

Robustness Evaluation of RO-PUFs

In this section we present two types of attacks performed to evaluate the robustness of the RO-PUF in previous works. In 2010, Ruhrmair et al. [RSS+10] attest that RO-PUF structure as proposed by Suh et al. [SD07] can be easily modeled and then it is attackable using modeling attack. Ruhrmair et al. [RSS+10] claim that even when the attacker could not eavesdrop all CRPs, the attack still easy to perform. They propose to apply a Quick Sort of randomly selected CRPs to predict the PUF responses. The authors demonstrate that this structure presents a 1% as a maximum prediction error rate using 14060 tries when performed on 256 RO-PUF. We can thus say that the RO-PUF is (99%, 14 060)-modelable.

In 2011, Merli et al. [MSSS11] present interesting results on the Electro-Magnetic (EM) attacks of the first proposed RO-PUF by Suh et al. [SD07] without using the masking technique. The EMA are a SCA technique. They can be classified as semi-invasive attacks or a non invasive ones. This depends, for example, on the ability of the attacker :

- to remain stable the device temperature during the whole acquisition process.
- to perform the analysis without unpackaging the FPGA or the ASIC platform.

The EM attack is applied for several RO basic architectures. First, Merli et al. [MSSS11] proposed to clone the RO PUF architecture using EM analysis. Later on, Bayon et al. [BBAF13] attacked RO-based TRNGs by locking the oscillator with an external oscillator.

The attacker can also build a model of the PUF and then break its security by deducing the PUF response for all introduced challenges. When the EM attack is applied for ring-oscillator PUFs (RO-PUFs), the attacker could :

– Extract the all ring-oscillator frequencies.
– Identify the position of each one on the die.
– Deduce the model of the attacked RO-PUF.

In their experiments, Merli et al. [MSSS11] consider a RO-PUF with 9 ring-oscillators. The experimental results show that the authors are able to extract all the ring-oscillators PUF and then to predict all CRPs. To remedy to such attacks, they propose to use the same masking technique proposed by Suh et al. [SD07] to avoid correlated bit responses.

1.4.1.3 Clock PUF

Recently, Yao et al. [YKL+13] proposed a novel delay PUF structure. It uses the clock network of a given IC to generate a PUF response. The Clock PUF compares the arrival times of selected clock signals to generate the PUF response. Figure 1.16 shows the proposed clock PUF circuit. To validate their proposed

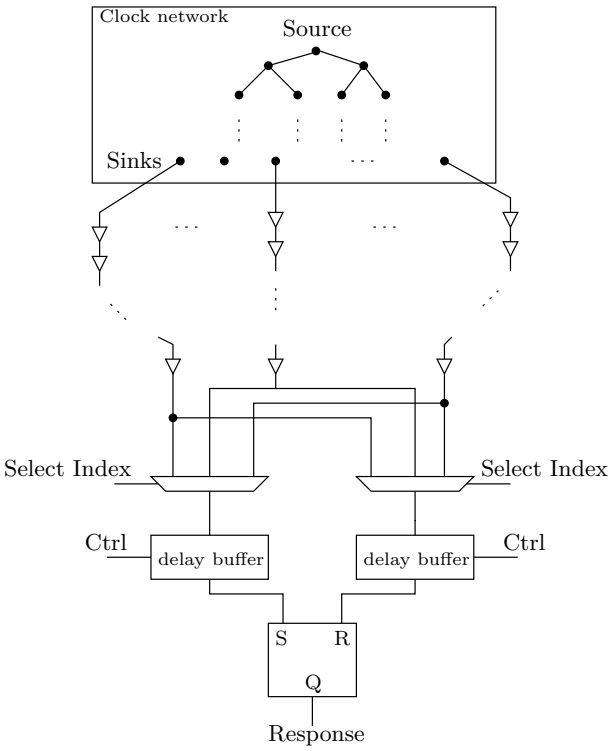


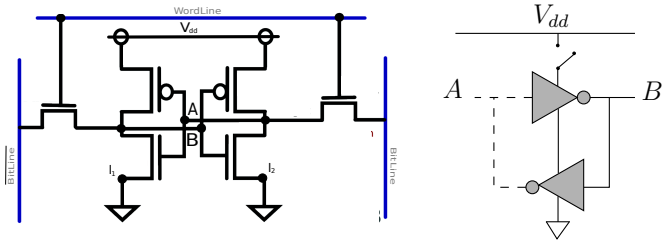
FIGURE 1.16 – Clock PUF structure.

structure, Yao et al. present a SPICE based performance evaluation results using a 45nm CMOS technology. Results show that the clock PUF presents an intra-distance percentage of 5.07% and an inter-distance percentage of 50.3% under nominal operating conditions. However, no studies on the vulnerabilities of this PUF structure have been presented yet.

1.4.2 Memory based PUFs

1.4.2.1 SRAM PUFs

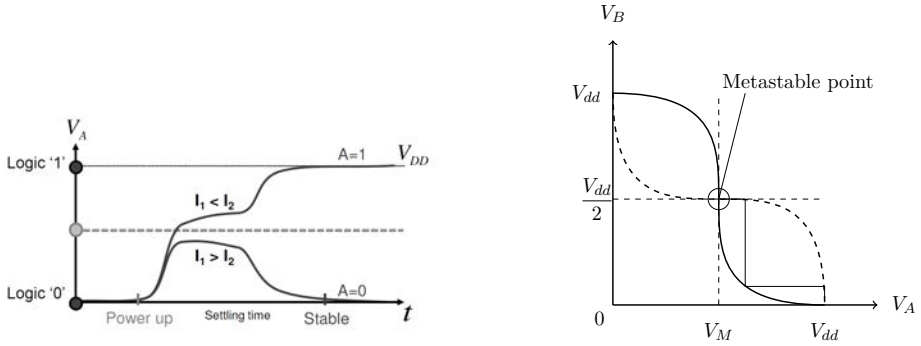
Six Transistors SRAM Cells



(a) SRAM cell CMOS circuit.

(b) SRAM cell logic circuit.

FIGURE 1.17 – Six transistors SRAM.



(a) SRAM cell voltage transfer curves [Mae12].

(b) SRAM cell power-up transient analysis.

FIGURE 1.18 – SRAM cell voltage transfer curves and power-up transient analysis.

Static Random Access Memory or SRAM is a type of semiconductor memory. It is based on bi-stable circuits. Typically, an SRAM cell is composed by six CMOS transistors as shown in Figure 1.17a. They are arranged as two cross-coupled inverters (Figure 1.17b) and two access switchers used to control the access to the storage cell during write and read operations. The main characteristic of an SRAM cell is the Static-Noise Margin (SNM). It can be defined as the minimum noise voltage to flip the cell state. Increasing the SNM results a more stable cell. However, the SNM is directly influenced by the threshold voltage of the transistors. The manufacturing process variation induces a difference in the threshold voltage of the transistors of the SRAM cell. If the inverters are

perfectly balanced, the SRAM cell starts in a "metastable state" which is in the middle of the voltage excursion (the metastable state is roughly at $V_{dd}/2$). As there is always a slight difference, the memory cell at power up will converge towards one of a stable state, i.e 0 volts or V_{dd} volts.

Figure 1.18a shows, by the voltage transfer curves, the three operating points of the two cross coupled inverters. Figure 1.18b shows that due to the positive feedback in the circuit, any small deviation from the metastable state is recognized ; this is immediately amplified by the circuit. Hence, one of the two stable states is reached randomly.

Performance Analysis : Experiments and Results

To guaranty an efficient behavior of SRAM under normal operation, SRAM cells are designed to have perfectly matched inverters. When powered up, different SRAM cells are initialized with different and random values due to process variations of cross-coupled inverters. However, when the mismatch between the two inverters is large, some SRAM cells tend to be initialized with the same value when powered up several times. These two properties constitute the first characteristic of PUFs. Based on these properties, Guajardo et al. [GKJST07] and Holcomb et al. [HBF07] simultaneously proposed the two first SRAM based PUFs. Both of them take advantage from the same concept of SRAM cells.

Guajardo et al. [GKJST07] proposed this solution for FPGA platforms to protect their IPs. Therefore, experiments have been realized on different memory blocks on different FPGAs. Results show that using four different SRAM blocks located into two different FPGAs, the maximum average intra-distance of a single memory block is less than 4% at normal environmental conditions. But, when varying the temperature, the intra-distance increases to attained almost 14% at $-20°C$. To evaluate the inter-distance variation of SRAM blocks, the authors collect 8190 bytes of power-up values derived from different SRAM blocks (and different FPGAs). The results show that the average inter-distance is of 49.97% which is very close to the ideal average of 50%.

Holcomb et al. [HBF07] [HBF09] proposed a SRAM PUFs for device identification and random number generation for RFID tags. Therefore, experiments have been carried out two different platforms. the first platform is composed of 8 SRAM chips. The second platform is a population of 3 micro-controllers. For the SRAM chips, the authors obtain an average inter-distance of 43.16% and an average intra-distance of 3.8%. An average inter-distance of 49.34% and an average intra-distance of 6.5% obtained for the embedded memories in micro-controllers

More recently, Selimis et al. [SKA+11] presented an evaluation of six transistors SRAM implemented in $90nm$ CMOS technology. The authors used seventeen ICs for experiments. Each one embed four SRAMs. When evaluated at normal environmental conditions ($V_{dd} = 1.2V$, $Temp = 20°C$), results show an average inter-distance around 50% and an average intra-distance bellow 4%. Varying the

temperature between $-40°C$ and $80°C$, the intra-distance is always bellow 19% ($Temp = -40°C$). However, the intra-distance variation is always around 6% when varying the supply voltage ±10% of V_{dd}

Also, Bohm et al. [BHP11] presented results for the evaluation of embedded SRAMs into micro-controllers. Therefore, experiments have been carried out on three NXP micro-controllers. Each one provides two SRAM blocks denoted *blockA* and *blockB*. Only the *blockA* of the micro-controller can be used as a PUF for generating a key or an identifier (ID). In fact, an average of 8% is noted for the intra-distance variation at normal environmental condition for the *blockA* which is better than 18%, obtained for the *blockB* memory. When varying the temperature to $0°C$ and then $80°C$, the average intra-distance for the *blockA* is 8.92% at $0°C$ and 8.22% at $80°C$. For *blockB*, we observe an average intra-distance of 29.40% at $0°C$ and 22.20% at $80°C$.

Robustness Evaluation of SRAM PUFs

Recently, Helfmeier et al. [HBNS13] attested that the SRAM PUFs can be cloned using SCA technique. The authors propose to use the Photonic Emission Analysis (PEA) to clone the SRAM PUFs. They are non-destructive and semi-invasive attacks. Helfmeier et al. [HBNS13] proposed to dynamically extract the contents of embedded memories to reproduce another one with exactly the same behavior. To do so, two steps are needed :

- SRAM characterization. In order to recover the state of the SRAM using the photonic emission of the transistors at the initialization of the SRAM, the author propose to capture the near infrared photonic emission of the device under tests (DUT) using electrical stimulation. This step can be speed up when we remove the backside package of DUT.
- Circuit edition. The authors proposed to perform a Focused Ion Beam (FIB) to reproduce the same response as the DUT using the emission image information collected on the SRAM characterization step.

Based on the proposed PEA technique, the authors prove that they are able to emulate the whole SRAM PUF behavior in few amount of time (e.g. 5 minutes to clone 16 bits). They propose two create a cloned device (SRAM) using two methods, non destructive FIB circuit editor, or destructive FIB circuit editor. On both solutions, the SRAM cloning operates. The non destructive solution proposes just to trim some transistors. However, on the destructive solution the bit-stable state characteristic disappears since the authors propose to disconnect some transistors.

1.4.2.2 Latch and Butterfly PUFs

In addition to SRAM PUFs there are other types of PUFs which are based on bi-stable circuit. They take advantage from mismatch between cross-coupled

devices to generate a random and static ID. They can be used instead of SRAM PUFs when SRAMs are not available on the FPGA platform. However, there is no scientific papers that propose attacks or that evaluate the robustness of both of the latch and the butterfly PUF structures presented hereafter.

Latch PUFs

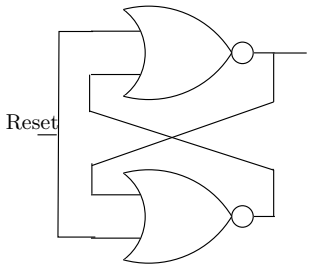


FIGURE 1.19 – Logical circuit of a latch PUF.

Su et al. [SHO07] propose a novel structure of PUF. It is composed of cross-coupled NOR-gates which constitute a simple SR latch (Figure 1.19). Both other sides of the latch are connected to a reset signal. In fact, both sides are initially pulled low. Therefore, the latch is initially forced into an unstable state. Then, when the reset signal is released, and depending on the mismatch between the two logic gates, the circuit will converge to one of the two stable states. The latch PUF behavior make it more interesting than SRAM PUFs since IDs can be generated at any time we want. Because it does not rely on a power up condition like SRAM PUFs. To evaluate this structure, experiments have been carried out of 19 test chips designed in 130nm CMOS technology. Results show that the circuit presents an intra-distance percentage of 3.04% and an inter-distance percentage of 50.55%.

Butterfly PUFs

Kumar et al. [KGM+08] present another based bi-stable circuit PUF. It is designed as two cross-coupled latches (Figure 1.20). By driving the preset signal of the latch 1 and the clear signal of the latch 2 by an excite input signal, we are able to force the circuit into an unstable state. When released, the circuit converges to a stable state depending on the mismatch between the two latches. To evaluate the proposed PUF structure, measurements are done using 36 Virtex-5 Xilinx FPGAs. The obtained measured intra-distance is bellow 0.6% under high temperature. However, the inter-distance is very close to 50% which is the ideal percentage.

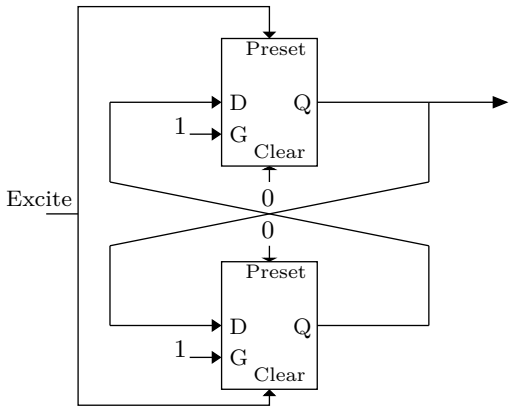


FIGURE 1.20 – Schematical circuit of a butterfly PUF.

The disadvantage of these two cross coupled structures is that we need hard routing constraints to be sure that we take advantage only from manufacturing process variation between the devices. However, this is very hard to perform in an FPGA platform since we have to use routed and placed cells.

1.4.3 Discussions

Table 1.1 summarizes the performance of presented intrinsic PUFs. According to the collected results, three main factors make the comparison of the different PUFs structures difficult :

1. The targeted devices use different CMOS technologies (e.g. $0.18\mu m$,90nm, 65nm, etc.). However, the performance of silicon PUFs is related to the process variation during manufacturing which is highly dependent on the device technology.
2. The different performance results (intra-die and inter-die) of the PUF structures presented in Table 1.1 are obtained using close but not identical metrics. Even when all the obtained results use Hamming computation based metrics, this is not sufficient to fairly compare the performance PUF structures.
3. The environmental test (e.g. noise, etc.) is not the same for all structures.

Hereafter, we present recently presented performance evaluation of a selection of silicon PUFs. The studies are supported bye the European Project "UNIQUE". Katzenbeisser et al. [KKR+12] propose to evaluate different silicon PUFs when they are embedded in the same ASIC platform using the same metrics and same environmental characteristics. Then a fair comparison of the PUF structure is possible. The experiments have been carried out 96 ASICs using the TSMC 65nm technology. They embed five silicon PUF structures (arbiter, RO, SRAM,

TABLE 1.1 – Overview of experimental results of intrinsic PUF structures in the literature.

PUF types	**Reference Paper**	**Env. Cond**	**Population (type/techno.)**	**Nb. of ch SRAM size**	**Intra-die eval.**	**Inter-die eval.**
Arbiter PUFs	Basic arbiter PUF [GLC+04]	Nominal	23 (FPGA/0.18μm)	100000	0.098 %	1,05 %
		67°C	23 (FPGA/0.18μm)	100000	0.3 %	1.05 %
	Basic arbiter PUF [LLG+04]	Nominal	37(ASIC/0.18μm)	10000	0.7 %	23 %
		67°C	37(ASIC/0.18μm)	10000	4.8 %	23 %
		Vdd	37(ASIC/0.18μm)	10000	3.7 %	23 %
	FF arbiter PUF [LLG+04]	Nominal	37(ASIC/0.18μm)	10000	2.2 %	40 %
RO PUFs	Single RO [GCvDD02]	Nominal	4(FPGA/0.18μm)	-	0.1 %	1 %
	RO PUF [SD07]	120°C, 1.08V	15(FPGA/90nm)	128	0.48	46.15 %
	RO PUF [MCMS10]	Nominal	125(FPGA/90nm)	511	0.86 %	47.13 %
		65°C	5(FPGA/90nm)	511	4 %	47.13 %
		0.096V	5(FPGA/90nm)	511	15 %	47.13 %
	Configurable RO PUF [MS09]	Nominal	5(FPGA/90nm)	255	0 %	44.1 %
		65°C	5(FPGA/90nm)	255	0 %	44.1 %
		Vdd -20 %	5(FPGA/90nm)	255	0 %	44.1 %
SRAM PUFs	SRAM PUF [GKJST07]	Nominal	2(FPGA/-)	8190	3.57 %	49.97 %
		−20°C	2(FPGA/-)	8190	13 %	49.97 %
	SRAM PUF [SKA+11]	Nominal	70(ASIC/90nm)	2048	<4 %	50 %
		−40°C	70(ASIC/90nm)	2048	<19 %	50 %
		+-10% Vdd	70(ASIC/90nm)	2048	<6 %	50 %
Latch PUF	Latch PUF [SHO08]	Nominal	19 (ASIC/0.13μm)	128	3.04 %	50.55 %
Butterfly PUF	Butterfly PUF [KGM+08]	80°C	36(FPGA/65nm)	-	<6 %	50 %

TABLE 1.2 – Experimental results of intrinsic PUF structures in the same ASIC platform.

PUF types	**Nb. of** instances	**Nb. of ch SRAM size**	**Response** Size	**Intra-die eval.**	**Inter-die eval.**
Arbiter PUF	256	2^{64}	2	<6%	<1%
RO PUF	16	2^{15}	2	<6%	<2%
SRAM PUF	4(8KB)	2^{11}	2^{32}	<7%	>80%
Flip-flop PUF	4(1KB)	2^{8}	2^{32}	<15%	very variable
Latch PUF	4(1KB)	2^{8}	2^{32}	<25%	

flip-flop, latch). Based on the same evaluation metrics, the authors propose to evaluate the structures on different ambient temperatures (−40°C to +85°C), supply voltages (±10% of the nominal 1.2 V) and noise levels (active core enabled and disabled). Table 1.2 recapitulates the mean performance results of the embedded PUFs under worst case environmental conditions. The latter changes depending on the PUF structure. For example for the arbiter PUF the worst steadiness results are obtained when we only increase the noise.

The authors redefine new metrics to evaluate the inter-die and the intra-die variations. They note that the **ideal** percentage of **intra-die variation** is **0%**. And the **ideal** percentage of **inter-die variation** is **100%**. The results show that the intra-die variation of the arbiter, the RO and the SRAM PUFs is acceptable (<7%). Then, they can be used even for critical applications using a lightweight error correcting schemes. However, the flip-flop and the latch PUFs present an intra-die variation of at most 15% and 25%, respectively which let them impractical in some applications.

The authors propose also inter-die variation studies under different environmental conditions. They show that the SRAM PUF presents the highest performance results (>80%). The arbiter and the RO PUFs present too low inter-die (uniqueness) performance (<2%). However, the latch and the flip-flop PUFs uniqueness performance are very dependent from the temperature variations which let them easy to attack.

Based on the literature results, we propose to compare the implementation and security characteristics of the studied PUFs (Table 1.3). We note that the SRAM PUFs are the easiest structures to implement ; they do not need any routing or placement constraints. However the arbiter PUFs are the most constraining in terms of routing and placement. The RO PUFs are the most vulnerable structures, they can be attacked both mathematically and physically. However they present better performance results than the bistable PUF structures as shown in Table 1.2.

The existing PUF structures present different performances which are important to evaluate with relevant metrics. the next section we detail the existing

TABLE 1.3 – Implementation and security characteristics of a selection of silicon PUFs.

PUF types	Impl. size	P & R const.	Vulnerabilities
Arbiter PUF	Small	High	Modeling attack
RO PUF	Small	Medium	Modeling attack EM attack
SRAM PUF	Medium	No	PE attack
Flip-flop PUF	Medium	High	
Latch PUF	Medium	High	

evaluation methods.

1.5 PUFs Evaluation Methods

Unfortunately, no standard statistical tests has been defined yet like the NIST [NIS12], the BSI [KS11] or FIPS [FIP01] ones used to evaluate the robustness of TRNG structures. To evaluate their PUFs structures, the authors often propose new methods and metrics that complicate the comparison process. A fair comparison of a given new proposition with a previous one would require the redesign and the re-evaluation of the latter. However, since the majority of intrinsic PUFs design need a lot of design carrying (Routing and placement constraints), the imperfect reproduction of the design may induce a faulted comparison. Therefore, even when authors reproduce the previous designs for comparison, this does not present a fair way to compare PUF structures. In this section we present a selection of different proposed method to evaluate PUF structure. We can divide them into two classes. The first one is based on the Hamming distance/weigh computations and the other one is based on statistical computations. The first paper oriented on methods and metrics for the evaluation of all intrinsic PUFs is Hori et al. [HYKS10] in 2010. Most of the proposed metrics are based on the statistical computations. The Hamming computations based metrics are presented by several authors when evaluating novel proposed structures.

As presented in Section 1.1.2, the most important properties that a PUF have to meet are the uniqueness and the reproducibility of the PUF responses. The randomness of the PUF is also pointed as an important property that a PUF have to verify to avoid biased responses. We consider that a PUF outputs a response R for a given challenge. The response length is L and we note b_l the l^{th} bit on the response R. The number of T tests are performed to evaluate the stability (intra-distance) of the PUF response either when changing the environmental conditions (temperature, voltage, etc.) or not. Also, the uniqueness (inter-distance) of the PUF response is studied when applying the same challenge for N different chips.

Hereafter, we detail the metrics based on Hamming distance/weigh and the one based on statistical computations.

1.5.1 Hamming Computation Based Metrics (Maiti et al. [MCMS10])

Almost all proposed PUF are evaluated using different Hamming distance/weight computations. All of them are nearly the same. As an example of the Hamming distance based metrics, we detail the ones proposed by Maiti et al. [MCMS10].

1.5.1.1 Randomness

A very simple way to evaluate the randomness of a PUF response is proposed. The uniformity of the response is evaluated by computing the Hamming weight of each PUF response R obtained when a challenge k is applied. The randomness of a response R_k is given by :

$$\mathsf{Rand_k} = \frac{1}{\mathsf{L}} \sum_{\mathsf{l}=1}^{\mathsf{L}} \mathsf{b_l} \times 100\%. \tag{1.1}$$

To be assessed as a uniform response, the metric result should to be close to 50%.

1.5.1.2 Steadiness

It is the evaluation of the stability (or the reliability) of the PUF response when varying operating conditions for a given chip n. A reference response R_n^{ref} is extracted at the nominal environmental condition. The same response R_n is extracted from the same chip n at different environmental conditions. Then, we compare the two responses using the Hamming distance between the measured responses and the reference. The average intra-device Hamming distance is used as an estimate of the steadiness property of the PUF on the chip n as defined bellow :

$$\mathsf{Stead_n} = \frac{1}{\mathsf{T}} \sum_{\mathsf{t}=1}^{\mathsf{T}} \frac{\mathsf{HD}(\mathsf{R_n^{ref}}, \mathsf{R_n^t})}{\mathsf{L}} \times 100\%. \tag{1.2}$$

The idea is to perform T times the same challenge for the same PUF at the same operating conditions and to compare the resulting response to the reference one extracted at the nominal environmental conditions. The same operation is repeated as much as the operating conditions are different. The obtained average Hamming distance is considered then as the reliability performance of the PUF. A **lower** value of the intra-device Hamming distance results in a **more steady**

PUF response.

1.5.1.3 Uniqueness

The average inter-device Hamming distance is used as an estimate of the PUF uniqueness property. Among the N existing chips, all possible pair-wise combinations are tested. Consider the two different chips u and v, and let us denote their responses by R_u and R_v, respectively. The proposed metric to evaluate the uniqueness as defined by Maiti et al. [MCMS10] is given by

$$\mathsf{Uniq} = \frac{2}{\mathsf{N}(\mathsf{N}-1)} \sum_{\mathsf{u}=1}^{\mathsf{N}-1} \sum_{\mathsf{v}=\mathsf{u}+1}^{\mathsf{N}} \frac{\mathsf{HD}(\mathsf{R_u}, \mathsf{R_v})}{\mathsf{L}} \times 100\%. \tag{1.3}$$

The **best** uniqueness performance is obtained when the inter-device Hamming distance is **close to 50%**.

1.5.2 Statistical Based Metrics (Hori et al. [HYKS10])

Until 2010, all PUFs evaluation methods are described when proposing a novel PUF structure. Several methods have been proposed, most of them are based on the Hamming computations. In 2010, a statistical method to evaluate the PUF performance is introduced by Hori et al. [HYKS10]. In addition to the randomness, the uniqueness and the steadiness properties, the authors propose to evaluate other performance indicators (e.g. diffuseness, correctness, correct ID, etc.) Most of the proposed metrics are based on statistical computations. The ideal performances are obtained when the metrics results are 100%. Hereafter, we presents the uniqueness, the steadiness and the randomness metrics as proposed by Hori et al. [HYKS10].

1.5.2.1 Randomness

To evaluate the balance of 0 and 1 in the PUF response on a chip n, the authors propose to compute :

1. The frequency of 1's when applying K different challenges and repeating each one T times for a selected chip n. We note p_n the computed value.
2. The min-entropy of the resulting probability.

This can be expressed as follow :

$$\mathsf{p_n} = \frac{1}{\mathsf{K.T.L}} \sum_{\mathsf{k}=1}^{\mathsf{K}} \sum_{\mathsf{t}=1}^{\mathsf{T}} \sum_{\mathsf{l}=1}^{\mathsf{L}} \mathsf{b_{n,k,t,l}}, \quad \mathsf{then}, \quad \mathsf{Rand_n} = -\mathsf{log_2}\ \mathsf{max}(\mathsf{p_n}, 1-\mathsf{p_n}). \tag{1.4}$$

We note that, compared with the Hamming Weight metric, this method takes into account the stability of the PUF response which makes it much more precise. It also evaluates the randomness of all possible applied challenges.

1.5.2.2 Steadiness

As defined by Hori et al. [HYKS10], the steadiness of a given PUF response is seen as the inverse of its error probability. We denote by $Sted_{n,k,l}$ the steadiness of a selected bit l on the chip n when applying the same challenge k ($Stead_{n,k,l} = 1 - Error_{prob}$). Therefore, the authors propose, first, to compute the probability to obtain 1 when applying the same challenge k for a selected bit l. Then, the error probability is obtained when computing the min-entropy of the resulting value. This can be expressed as :

$$\mathsf{p_{n,k,l}} = \frac{1}{\mathsf{T}} \sum_{\mathsf{t=1}}^{\mathsf{T}} \mathsf{b_{n,k,l,t}}, \quad \mathsf{then}, \quad \mathsf{Stead_{n,k,l}} = 1 + \mathsf{log_2}\ \mathsf{max}(\mathsf{p_{n,k,l}}, 1 - \mathsf{p_{n,k,l}}).$$

Based on the previous computations, the device steadiness $Stead_n$ is defined as the average of $Stead_{n,k,l}$ as follows :

$$\mathsf{Stead_n} = \frac{1}{\mathsf{K.L}} \sum_{\mathsf{k=1}}^{\mathsf{K}} \sum_{\mathsf{l=1}}^{\mathsf{L}} \mathsf{S_{n,k,l}}. \tag{1.5}$$

Comparing to the Hamming based steadiness metric, we do not need a reference response and the stability of the PUF can be studied even at the same environmental condition.

1.5.2.3 Uniqueness

As proposed by Maiti et al. [MCMS10], Hamming computations are used for the uniqueness evaluation. The Hamming distance of all the possible device combinations is considered to evaluate the uniqueness of a PUF response. The authors propose to proceed bit by bit when varying the applied challenge. The mean device uniqueness $Uniq$ is given by

$$\mathsf{Uniq} = \frac{4}{\mathsf{K.L.N^2}} \sum_{\mathsf{k=1}}^{\mathsf{K}} \sum_{\mathsf{l=1}}^{\mathsf{L}} \sum_{\mathsf{u=1}}^{\mathsf{N-1}} \sum_{\mathsf{v=u+1}}^{\mathsf{N}} (\mathsf{b_{u,k,l}} \mathsf{b_{v,k,l}}). \tag{1.6}$$

1.6 Conclusion

In this chapter, we described a general background about PUF concept, application, classification and performance and robustness evaluation.

From a proposed classification of PUFs into intrinsic and non intrinsic PUFs as given by Maes et al. [Mae12], we mainly focused on intrinsic PUFs since they are advantageous regarding security and cost-efficiency. However, most of them, especially delay PUFs, need many implementation constraints and can be attackable using modeling or side-Channel attacks. Therefore, in this thesis, we studied two novel structures of delay PUFs with fewer constraints for routing and placement (Loop PUF) and more secure against EM attacks (TERO PUF). We also present a method to use them for cryptographic key generation and authentication purposes.

In this chapter, we presented an overview of existing performance evaluation methods. They can be divided into two classes : one based on the Hamming computations and another based on statistical computations. The latter is certainly the most accurate, but both of them need a huge number of tests to evaluate the PUFs. In this work we propose a statistical method to evaluate delay PUFs at design stage. The proposed method takes advantage of simulations of the physical values (i.e. the delays or frequencies). One interest is that it needs less tries to evaluate the PUF performance than classical methods.

One main contribution of this thesis is to propose novel PUF structures with better performances, high reliability, design easiness and robustness against EM attacks.

Chapitre 2

Loop PUF

The topic addressed in this chapter is related to the architecture of a novel structure of silicon delay PUFs. The objectives of this new PUF is, first, to increase significantly the reliability of the PUF response, and second, to provide a design methodology that avoids stringent constraints in the backend design stage for either FPGA or ASIC targets. This PUF is composed of identical controllable delay chains and an inverter organized in a loop to obtain a controllable ring oscillator. The proposed structure, referred to as "loop PUF", response is generated from sequential comparisons of oscillation frequencies for different control words. The reliability is enhanced by considering multiple oscillation measurements. This is equivalent to repeat so multiple tests of the arbiter PUF. Also, as the measurement is not differential, as more than two lines can be connected serially, this greatly increases the number of challenges. Compared to RO-PUF, only one oscillator runs, thus avoiding locking phenomenon and allowing the designer to surround the structure with shielding. The backend design is also easier as it consists in merely a copying and pasting a first delay chain without any line crossing or specific place and route constraints for other delay chains.

The loop PUF has been compared to the arbiter PUF on two CMOS 65nm ASIC and FPGA platforms. The performance analysis was performed under different environmental conditions, allowing us to study :

– the impact of the CMOS 65nm technology on delay PUFs when designed in ASIC or FPGAs.
– the comparison between arbiter and loop PUF when designed for the same platform.

2.1 Loop PUF

The proposed PUF, referred to as "loop PUF", is an intrinsic delay based PUF. The traditional approach to design them is based on differential delay measurement comparisons of basic delay elements. The loop PUF compares multiple identical delay chains implemented sequentially. The main benefits of the proposed PUF structure are :

- The easiness to design it either on ASICs or on FPGAs. It consists on the copy/paste of an already placed and routed delay chain. In fact, there is no need for routing constraints inside a delay chain.
- The easiness to enhance the reliability of the PUF response. When enlarging the delay measurement, then increasing the number of oscillations, the reliability of the PUF response is enhanced without any post-processing schemes.
- The huge number of possible challenges because of the non differential structure of the loop PUF.

However, due to the needed sequential measurements for the PUF response computations, the loop PUF is slow. It needs around tens of milliseconds to output the PUF response which remains acceptable in some applications.

Its structure can be divided into two parts as shown in Figure 2.1. The first part presents the data path part. It is the most important part since we exploit its process variation to generate the PUF response. The second part includes the control of the data path part. It is necessary either to manage challenges or to compute the PUF response (ID). In what follows we provide a detailed description of the parts.

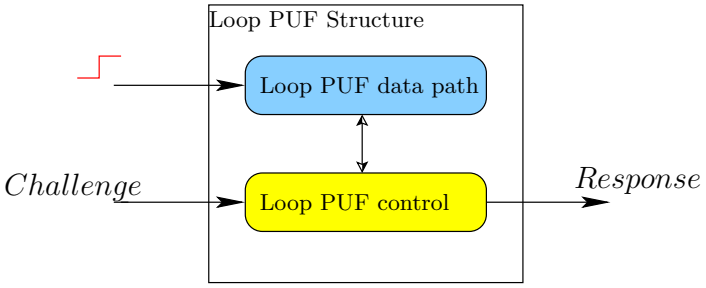


FIGURE 2.1 – The Loop PUF structure.

2.1.1 Data path part

The basic element in the data path part is a controlled delay element as shown in Figure 2.2. The concatenation of M delay elements forms a delay chain. To avoid routing constraints, N identical delay chains are sequentially concatenated and closed by an inverter to form an oscillating loop. Each delay chain is

controlled by a control word. The concatenation of the N control words forms the PUF challenge. Then, the loop PUF data path can be seen as a single controlled ring oscillator. A step input triggers all delay chains sequentially. Then, the ring oscillator output is used to measure the oscillation frequency by the loop PUF control part. Theoretically, identically controlled delay elements must have the same delay. However, due to the manufacturing process variation, the similar involved transistors don't have the same signal delay propagation. The proposed loop PUF structure takes advantage from this process variation to output the PUF response. The oscillation frequency of the PUF depends on the order on which the control bits are applied. Then, the use of a differential measurement of two measured frequencies when switching their applied control words, gives us a non null value which can be used to generate the PUF response.

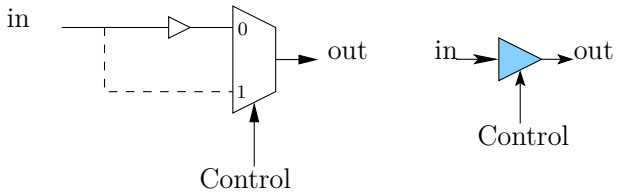


FIGURE 2.2 – Structure of a basic delay element.

Compared to the RO PUF [SD07], the loop PUF has only one oscillator and there is no delay chain pairing (N can be greater than 2). The noise introduced to the loop PUF impacts all delay chains and the local cross-coupling is limited as there is only one oscillator. Compared to the arbiter PUF [GCvDD02], the structure of the loop PUF is simpler as there is no need to cross wires in the delay elements and extra logic to balance the two chains as in [MKD10].

The only design constraint imposed to build the loop PUF is to duplicate the delay chain N times with a faithful reproduction of the placing and routing. This constraint is quite easy to meet in ASIC. In FPGA, we can be doubtful as the routing structure is unknown and well protected by some FPGA manufacturers. Our experiments conducted in the next chapter show that it is possible to duplicate small structures such as delay chains in Xilinx FPGAs. Figure 2.3 shows that the delay chain has no internal place and route constraints. Delays between delay elements can be different. The only requirement is a perfect N times duplication of a reference delay chain.

Hereafter, we present the control part of the loop PUF structure which is in charge of the extraction of the manufacturing process variation.

2.1.2 Control part

The loop PUF controller is in charge of extracting the result that is either an intrinsic key or the response of a Challenge-Response Pair (CRP) authentication.

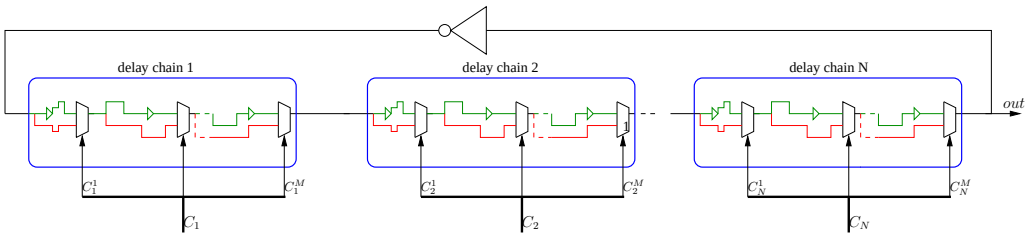


FIGURE 2.3 – Loop PUF datapath.

To do so it has :

1. to drive the loop PUF data path by a set of N control words C_i,
2. to measure the corresponding frequency or period,
3. to compare resulting periods (delays).

For a given set of control words called "Challenge" $C_1, ..., C_N$, the controller applies different combinations of the control words. Then, the controller measures the loop oscillation frequency f or the delay d when a combination of control words $C_1, ..., C_N$ is applied. For instance, if we consider a loop PUF structure composed of three delay chains $N = 3$. The challenge inputs the loop PUF with three control words $C_1C_2C_3$, let's say ABZ. Then the loop PUF controller makes 3 rotations of ABZ and measures the delay for each.

The result should remain the same for all permutations of C_i if the delay chains are perfectly balanced. But in physical devices there is a slight delay discrepancy because of CMOS variability which is exploited to build intrinsic silicon PUFs.

As an example we can consider $N = 2$ and the delay element j illustrated in Figure 2.4.

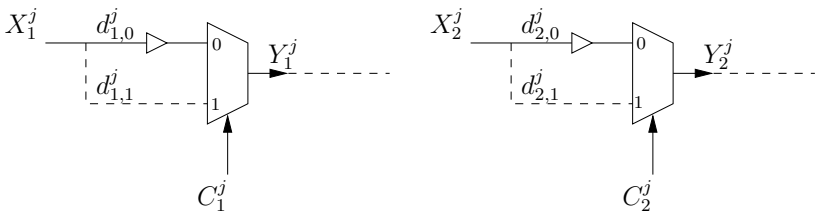


FIGURE 2.4 – Delay element j for two delay chains.

If an oscillation period measurement is done with the combination $C_1^j = 0$ and $C_2^j = 1$, then with $C_1^j = 1$ and $C_2^j = 0$, the difference of the two measured delays is

$$D^j = (d_{1,0}^j + d_{2,1}^j) - (d_{1,1}^j + d_{2,0}^j) \neq 0.$$

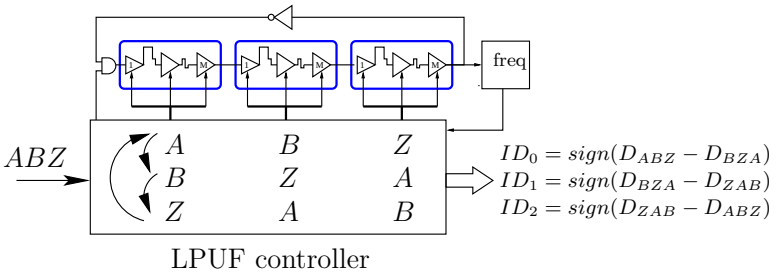


FIGURE 2.5 – Loop PUF control, example with $N = 3$.

The delay difference D^j should be ideally equal to zero, but this is never the case because of the process dispersion.

Several methods can be used to extract the PUF response using the oscillation delay of the loop. In our case, we are interested on the pair wise comparisons of the oscillation delays. We propose to rotate the applied control word and then to measure oscillation delays of the loop at each control word combination. Only N rotations are considered. After rotating challenges and doing measurements, we have to compare oscillation periods to generate the PUF response. The sign of each comparison correspond to a one bit response. The PUF response length is, indeed, correlated with the number of delay chains since only N comparisons are carried out.

For instance, if $N = 2$ and we consider the control words C_1 and C_2, the PUF response (ID) can be expressed by

$$\begin{aligned} ID &= sign(D_{C_1C_2} - D_{C_2C_1}) \\ &= sign\left(\sum_{j=1}^{M}(d^j_{1,C_1^j} + d^j_{2,C_2^j}) - (d^j_{1,C_2^j} + d^j_{2,C_1^j})\right). \end{aligned}$$

If the frequency is measured instead of the time, the same equations hold by using the frequency difference rather than the delay difference. This is illustrated in Figure 2.5 with N=3. The challenge inputs the loop PUF with three control words $C_1C_2C_3$, says ABZ. Therefore the loop PUF controller makes 3 rotations of ABZ and measures the delay for each rotation. The loop PUF returns an ID code of 3 bits.

In this case the 3 bits of the loop PUF identity are expressed by

$$\begin{aligned} ID_0 &= sign\,(D_{ABZ} - D_{BZA}) \\ &= sign\left(\sum_{j=1}^{M}(d^j_{1,A^j} + d^j_{2,B^j} + d^j_{3,Z^j}) - (d^j_{1,B^j} + d^j_{2,Z^j} + d^j_{3,A^j})\right), \\ ID_1 &= sign\,(D_{BZA} - D_{ZAB}) \\ &= sign\left(\sum_{j=1}^{M}(d^j_{1,B^j} + d^j_{2,Z^j} + d^j_{3,A^j}) - (d^j_{1,Z^j} + d^j_{2,A^j} + d^j_{3,B^j})\right), \\ ID_2 &= sign\,(D_{ZAB} - D_{ABZ}) \\ &= sign\left(\sum_{j=1}^{M}(d^j_{1,Z^j} + d^j_{2,A^j} + d^j_{3,B^j}) - (d^j_{1,A^j} + d^j_{2,B^j} + d^j_{3,Z^j})\right). \end{aligned} \tag{2.1}$$

We have to note that the inconvenience of this structure is that we need more time than required for the arbiter PUF measurement because of sequentially measurements of delay chains. However, the greater measurement time (a few ms) could be largely acceptable for many applications (*e.g.* generation of cryptographic keys).

Only by selecting appropriate control word, the steadiness of the loop PUF can be enhanced. Hereafter, we show how much is the impact of the choice of the control word on the reliability of the PUF response.

Choice of Control Word

Ideally all the control words should be different from each other in order to have the maximum discrimination between two tries. In Equation (2.1), we can see that the control words, $A^jB^jZ^j$, determine the number of delays which contribute to the IDs. For instance if $A^j = B^j \neq Z^j$, the two delays d^j_2 and d^j_3 are used to calculate ID_0. The reliability is enhanced if more delays are used as the variance of the resulting distribution increases proportionally to the number of delays. The difference between two control words is expressed by the Hamming distance H.

$$H = \sum_{i=1}^{N}\sum_{i'>i}^{N} HW(C_i \oplus C_{i'}),$$

where HW is the Hamming Weight function of C_i XOR $C_{i'}$. As H should be maximal, it can be shown that for $M = 1$ (words of one bit), the H maximum H_{max} is given by this formula :

TABLE 2.1 – Number of challenges.

		M									
		2	3	4	5	6	7	8	10	12	16
$Ch_{arbiter}$		4	8	16	32	64	128	256	1024	4096	65536
$Ch_{loopPUF}$	$N=2$	4	13	40	121	364	1093	3280	29524	∼250K	∼21M
	$N=3$	4	44	360	2680	19244	∼130K	∼1M	∼45M	∼2G	∼5000G

$$\begin{aligned} N\ odd &\Rightarrow H_{max} = \tfrac{(N^2-1)}{4}. \\ N\ even &\Rightarrow H_{max} = \tfrac{N^2}{4}. \end{aligned} \tag{2.2}$$

$$\begin{aligned} D_{(00,01)} - D_{(01,00)} &= ((d^1_{1,0} + d^1_{2,0}) - (d^1_{1,0} + d^1_{2,0})) + ((d^2_{1,0} + d^2_{2,1}) - (d^2_{1,1} + d^2_{2,0})) \\ &= ((d^2_{1,0} + d^2_{2,1}) - (d^2_{1,1} + d^2_{2,0})) \\ &= ((d^1_{1,1} + d^1_{2,1}) - (d^1_{1,1} + d^1_{2,1})) + ((d^2_{1,0} + d^2_{2,1}) - (d^2_{1,1} + d^2_{2,0})) \\ &= D_{(10,11)} - D_{(11,10)} \end{aligned} \tag{2.3}$$

In addition to the requirement of having H maximum, there is another constraint which is needed to avoid equivalent control words. For instance if $N = 2$ and $M = 3$, the ID obtained from the challenge $(0, 1)$ is the same as $(2, 3)$, $(4, 5)$ and $(6, 7)$. Equation (2.3) with $N = 2$ and $M = 2$ shows that the challenge $(0, 1)$ is equivalent to $(2, 3)$.

The constraint to avoid equivalent challenges can be formalized by

$$\forall j \in [1, M] \ \prod_{i=1}^{N} C_i^j = 0. \tag{2.4}$$

Even with this constraint the number of possible challenges is much greater with regards to the arbiter PUF. The number of challenges for an arbiter PUF having M elements is 2^M, whereas the loop PUF has a total of 2^{NM} challenges minus the combination which does not meet the condition expressed in Equation (2.4). Table 2.1 shows the maximum number of possible challenges for $N = 2$, $N = 3$ and for different values of M.

A minimum number of challenges have to be chosen to generate an ID with nb_{bits} number of bits. For instance to obtain an ID of **64-bit** with $N = 3$ using the rotations on control words, the number of challenges is :

$$\left\lceil \frac{64}{\log_2 3!} \right\rceil = 26\ challenges. \tag{2.5}$$

Hereafter, we propose to detail the implementation strategy and to evaluate the performance of the loop PUF structure when designed on different platforms.

2.2 Delay PUFs on CMOS 65nm technology : ASIC and FPGA

In this section we present the implementation details and the experimental results of two delay PUFs. The same design is tested using the same technology in two different platforms. Hereafter, we present the studied IP and the performance of the PUFs that includes.

2.2.1 PUF IP Specification

The rational behind of the design is, mainly, to collect data to analyse the PUFs performances. The challenge is to implement as much as possible the same PUF structure on different location on the circuit to evaluate its performance according to its placement on the circuit. We would like to implement two different types of delay PUFs in order to compare their performance when designed in the same conditions and using the same CMOS technology.

2.2.1.1 Design Requirements

The PUF Module
We select two intrinsic delay PUF structures to design on both platforms ASIC and FPGA.

1. The arbiter PUF as proposed by Majzoubi et al. [MKD10].
2. The loop PUF as proposed by Cherif et al. [CDGB12].

The selected structures are based on delay chains. To fairly compare the PUF performances, we proposed a design named PUF_{mix} where same delay chains are used for both structures. Several instances of the PUF module are implemented on the same platform.

The Control Part
The selection and the activation of the desired PUF module is provided by the control of the PUF IP. Several arbiter PUFs can be activated at the same time. However, only one loop PUF can be active at one time. The control part is also in charge of the computation of the oscillation frequency of the loop PUF in order to deduce its response. The measurement of the oscillation frequency of the loop PUF is performed by computing the number O of the observed periods of the *clock* (T_{clk}) during a fixed oscillation window of $2^t \times T_{puf}$, with t an input value (Figure 2.6). We can denote $T_{puf} = O \times T_{clk} \times 2^{-t}$.

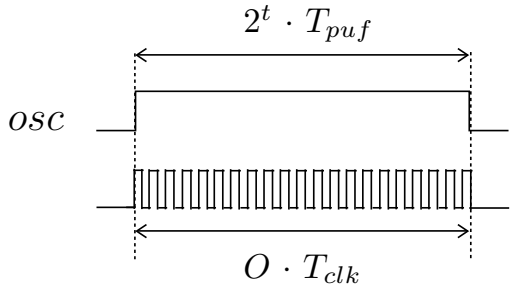


FIGURE 2.6 – Measurement window of the loop PUF oscillation frequency.

TABLE 2.2 – Communication interface (I/0).

`Clock`	in	Clock system.
`Reset`	in	Reset the components at low level.
`Address`	in	(7bits) Register address.
`Data_in`	in	(8bits) Input data.
`Data_out`	out	(8bits) Output data.

If we consider that the number of the observed oscillation O is limited to 16 $bits$, then $t \leq 16 + log_2(F_{puf}/F_{clk})$. The oscillation frequency of the loop PUF when the $F_{clk} = 25MHz$ is around $100MHz$ based on electrical simulation (Spectre). This means that $t_{max} = 18$ and then we need at most 5bits to define t at $F_{clk} = 25MHz$.

The Interface Part

To transfer the measurement data out of the IP, we require a communication interface. We prefer to use a standardized interface for easy integration with other components. A Universal Asynchronous Receiver/Transmitter (UART) was selected. The configuration and the communication with the PUF IP is entirely ensured using the UART interface (Table 2.2).

2.2.1.2 Top-Level Architecture

Figure 2.7 indicates the top-level architecture of the PUF IP. All data communication pass through the interface part. The communication between the PUF modules and the controller uses a large number of multiplexers and demultiplexers.

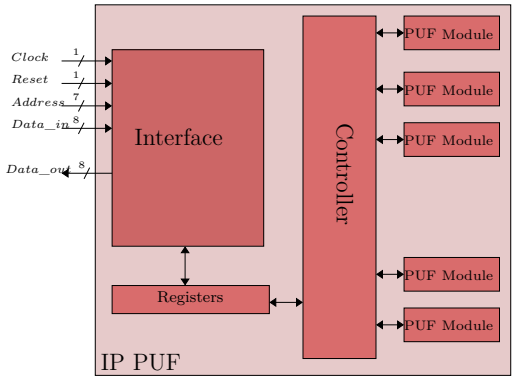


FIGURE 2.7 – Top-level architecture of the PUF IP.

2.2.2 Design Under Tests : The PUF Module

The PUF module is a mixed PUF design called PUF_{mix}. It is composed of two PUF structures which uses the same delay chains in order to perform a fair performance comparison. The PUF_{mix} design is composed of three independent PUFs. Two arbiter PUFs are designed according to the improved construction given by Majzoobi et al. [MKD10]. And, one loop PUF based on the construction from Cherif et al. [CDGB12].

2.2.2.1 Architecture of the Arbiter PUF

The arbiter PUF response is directly related to the result of the race between two identical paths. The first structure proposed by Gassend et al. [GCvDD02] is made up of M identical switchers structured as a mini crossbar 2x2 and an arbiter represented by a flip-flop at the end (See Section 1.4.1.1 and Figure 2.8).

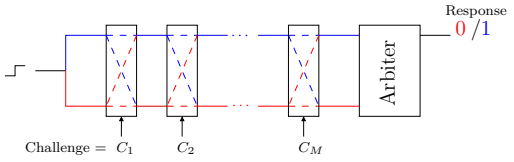


FIGURE 2.8 – Arbiter PUF structure.

To ensure that the delay difference takes advantage only from CMOS variation, routing constraints are needed to make two identical cross coupled delay lines. In order to reduce the routing constraints and the potential imbalance between the two lines of the arbiter PUF, Majzoobi et al. [MKD10] propose an arbiter PUF based on two identical and parallel similarly controlled delay chains composed of M identical controllable delay elements. The delay element can be composed of a buffer and a multiplexer as shown in Figure 2.2. Figure 2.9 illus-

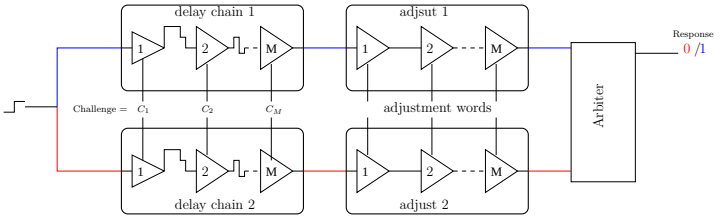


FIGURE 2.9 – Improved arbiter PUF structure.

trates the arbiter PUF proposed by [MKD10] where adjustment lines have been inserted to avoid the natural imbalance of the arbiter PUF. For presentation purposes, we replace each basic controllable delay element by a triangle.

2.2.2.2 Architecture of the Loop PUF

Even with the design proposed by Majzoobi et al. [MKD10], the arbiter PUF still needs routing constraints before and after the delay chains. In order to avoid these constraints, we [CDGB12] proposed a delay PUF referred to as "loop PUF". As the arbiter PUF, the loop PUF is based on N identical delay chains ($N \geq 2$). The delay chains are connected serially and do not require routing constraints, except a mere copy/paste of each delay chain. When closed by an inverter, this structure forms a loop which oscillates, as a single ring oscillator. The loop PUF architecture is illustrated in Figure 2.10. Each delay chain is controlled independently.

The loop PUF response is derived from the difference between measured frequencies when applying different permutations of the same set of control words $C_1, ..., C_N$. For a given set of control words, the controller applies different combinations of the control words and measures the oscillation frequency of the loop. The result should remain the same for all permutations of C_i if the delay chains are perfectly balanced. However, because of the CMOS variability in physical devices, the measured frequencies are slightly different.

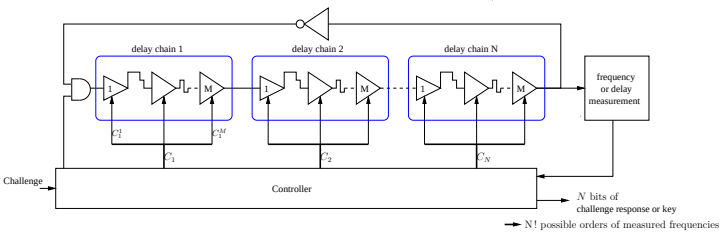


FIGURE 2.10 – Loop PUF structure.

2.2.2.3 The PUF_{mix} Structure

Figure 2.11 shows the PUF_{mix} structure designed on both FPGA and ASIC platforms. Four delay chains are used on the PUF_{mix} design to make 3 independent PUFs :

- Arbiter PUF #1 (uses the two upper delay chains).
- Arbiter PUF #2 (uses the two bottom delay chains).
- Loop PUF (uses the four delay chains).

Each delay chain in composed of M=16 basic delay elements. At the end of each chain, we use a buffer to equilibrate the end charges of the four chains. A multiplexer is used before each delay to select the operating mode of the design (arbiter or loop PUF) depending on the *aorl_puf* signal. Using four delay chains, the loop PUF generates 4 delays which can be sorted by external controller to generate a 4-bit response. However, each arbiter PUF generates a one-bit response.

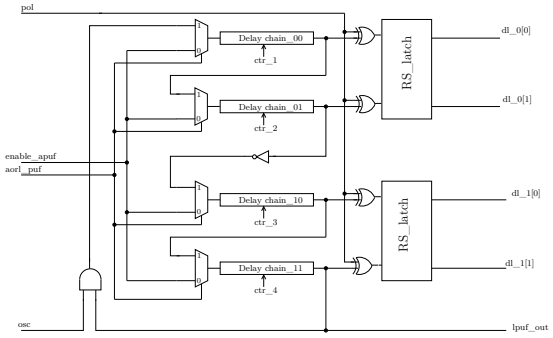


FIGURE 2.11 – PUF_{mix} design.

The PUF_{mix} Interface

Table 2.3 shows the input/output signals of the PUF_{mix} design.

The user can manage these inputs/outputs signals through the interface and the controller using defined registers.

2.2.3 Platforms Under Tests

The PUF IP is implemented in two platforms, ASIC and FPGA. Both of them use the CMOS 65nm technology. Implementation details are presented in the following sub-subsections.

2.2.3.1 ASIC Implementation Details

Designing an ASIC is a very complex process with a minimal margin of error. The probability of failure should be minimized. It should run at first time since

TABLE 2.3 – PUF_{mix} interface signal description

Signal name	Type	Bit size	Description
`pol`	in	1	'1' : Simpling at the falling edge. '0' : Simpling at the rising edge.
`enable_apuf`	in	1	A pulse signal to boost the arbiter PUF.
`aorl_puf`	in	1	select the PUF_{mix} mode. '0' : Arbiter PUF ; '1' : Loop PUF.
`osc`	in	1	'1' : Enable the loop PUF oscillation. '0' : Desable the loop PUF oscillation.
`ctr_1`	in	16	The control word of the delay chain 1.
`ctr_2`	in	16	The control word of the delay chain 2.
`ctr_3`	in	16	The control word of the delay chain 3.
`ctr_4`	in	16	The control word of the delay chain 4.
`dl_0[0]`	out	1	The output Q of the first latch (first arbiter PUF).
`dl_0[1]`	out	1	The output $\bar{Q}$ of the first latch (first arbiter PUF).
`dl_1[0]`	out	1	the output Q of the second latch (second arbiter PUF).
`dl_1[1]`	out	1	The output $\bar{Q}$ of the second latch (second arbiter PUF).
`lpuf_out`	out	1	Output of the loop PUF .

we have a single opportunity to produce an ASIC in the project.

These two constraints lead to the following choices :

- To minimize the risk of failure, the design is simulated before manufacturing.
- The silicon area for the implementation of instances is minimized taking into account the devoted budget.
- The whole design is synthesized using the ST-standard cells library for the 65nm technology. However, all placements and routing processes are done manually.

In order to evaluate the intra-device characteristics of the PUFs, the ASIC design embeds 49 PUF_{mix} as shown in Figure 2.12.

The 49 PUF_{mix} are placed on a 7×7 matrix. Each matrix cell contains one PUF_{mix}. 18 devices have been manufactured to evaluate the inter-device characteristics of the PUFs. The ASIC PUF_{mix} implementation uses 215 standard cells (343 gates). In fact, all output signals are isolated from the I/O pins using a buffer in order to keep equilibrate timing performances for the arbiter PUF. Hence, each PUF_{mix} occupies $50.8\mu\text{m} \times 44.28\mu\text{m} = 2249.424\mu\text{m}^2$ in the ASIC. Identical routed delay chains are designed using an automatic script in order to ensure identical paths (See Figure 2.13). Particular place and route efforts are done designing the *RS_latches* using 2 NAND gates to obtain an equilibrated feedback wires.

Each device is tested on a prototype board shown in Figure 2.14. This test board contains external pins to control the power supply for the ASIC core, in order to evaluate the reliability of the PUF under varying supply voltages. The

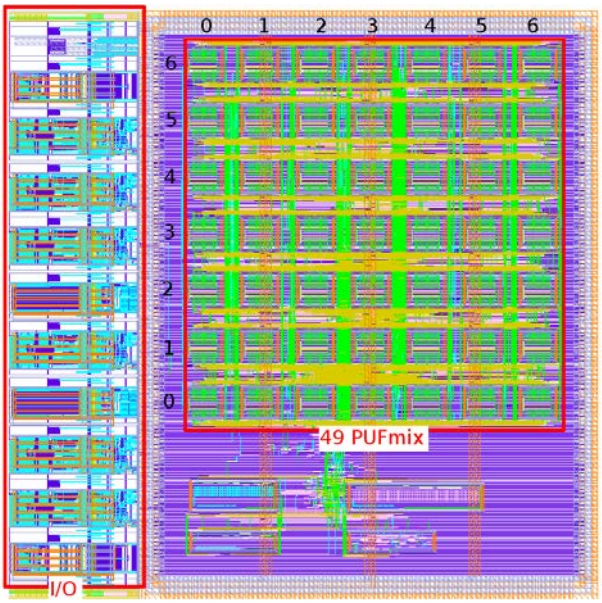


FIGURE 2.12 – ASIC layout.

communication with the device is done via an UART module which has its own clock and Power supply.

2.2.3.2 FPGA Implementation Details

A 168 PUF_{mix} design has been implemented on a Xilinx Virtex-5 vlx50t FPGA embedded on a Digilent Genesys development board. To obtain such a circuit, specific methodology has been used to achieve the two main constraints :

- The four delay chains have to be exactly similar in terms of resource placement and routing.
- The PUFs have to be cloned.

Achieving these constraints is not possible at RTL level only. It is also necessary to apply a methodology with specific Xilinx objects :

- *Hard Macro* : It is a placed and routed design part, which does not contain any input/output buffers (IOB). A *hard macro* can be instantiated more than once in a circuit. Each instance is a clone of the original module reproducing its placement and routing. *Hard macros* are stored in NMC files, a Xilinx file format is quite similar to *NCD* files, already used to describe classic netlists. Custom scripts have been developed to automatically generate *hard macros*.
- *Primitive instantiation* : We can include some Xilinx specific primitives in HDL code to infer for example LUTs, and define their truth table and inputs mapping. Those primitives are described in the Virtex-5 Libraries

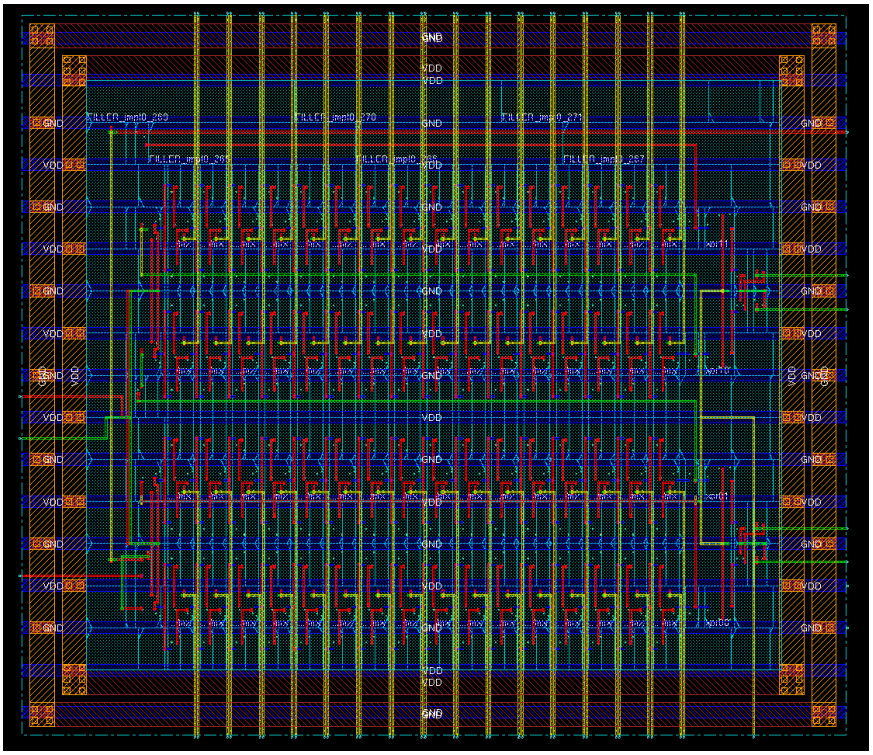


FIGURE 2.13 – PUF_{mix} layout.

Guide for HDL Designs [Xil].

– *Xilinx constraints* : Xilinx tools offer the possibility to attach to a particular design some constraints and attributes which affect the implementation like logical, physical or mapping constraints. Undesired place or route optimizations can also be forbidden.

Then the corresponding steps necessary to meet the Place/Route constraints are described below :

1. A primitive instantiation technique is first used to design a delay element chain where each element is a LUT which is configured as a MUX21. Elements are manually placed from left to right, on a same slice row. Manual placement is done taking advantage of design constraints that Xilinx provide us, like "LOC", "RLOC". Others constraints to prevent Xilinx design flow tools from making undesired optimizations are also added, like "SAVE NET FLAG" or LOCK_PINS". The two first LUT inputs are logically connected to the previous element output, while the 3^{th} input is reserved to be connected to one control signal (ctr). Virtex-5 slices consist of four LUTs, so one chain of 16 elements occupies four slices. After one chain is designed, our custom scripts are computed, and a "chain" Hardmacro is created, preserving its placement and routing.

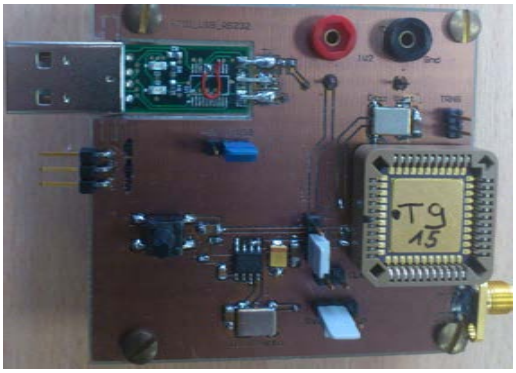


FIGURE 2.14 – Test board.

2. Then, chain Hardmacros are instantiated on successive rows. Additional logic operators used for both arbiter and loop PUF are manually placed. Also, routing is not constrained, so differences may exist between the two arbiter PUFs. Next, a PUF_{mix} Hardmacro is created.
3. Finally, all PUF_{mix} Hardmacro instances are manually placed on the FPGA matrix.

Every instance occupies 24 slices, which is about 0.4% of slice resources. The final Layout is shown in Figure 2.15.

Figure 2.15 shows that some columns are not usable for PUF_{mix} as it requires 4 adjacent columns whose slice type is M, L, L, L (M=Memory ; L=Logic), respectively. As for some columns the sequence is M, L, M, L, hence the PUF_{mix} cannot be placed here.

In order to fairly compare the PUF_{mix} performance when designed in both platforms, only the first 49 PUF_{mix} are considered on the FPGA.

2.2.4 Experimental Results

The goal of the experiments is to characterize the performance of the PUF_{mix} structure when designed in both ASIC and FPGA platform. We start first by the intra-device evaluation to compare the PUF_{mix} performance on FPGA to its performance on ASIC. Then, we evaluate the inter-ASIC performance for the PUF_{mix} design to compare the loop and the arbiter PUF characteristics under varying operating conditions.

2.2.4.1 Experiment Strategy and Setup

To compare the PUFs performance when designed in different platforms ASIC and FPGA, the PUFs are tested under nominal operating conditions. However, to determine the behavior of the different PUFs structures on ASIC, in particular

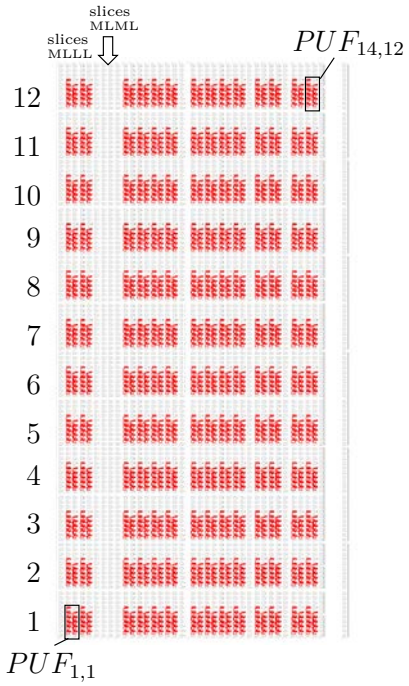
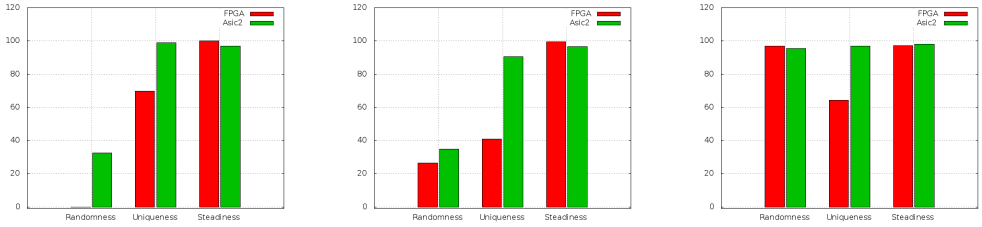


FIGURE 2.15 – Layout of FPGA design where 168 PUF_{mix} Hardmacros are instantiated.

regarding their reliability, they are tested under varying operating conditions. We use a cooled incubator to control temperature ranges of $0°C$ to $70°C$. An external voltage generator is used to vary the core supply voltage : V_{dd}= 1.02V ... 1.32V. At all considered conditions, all PUF_{mix} structures on all 18 ASICs are evaluated for a selected set of challenges. In all our experiments we use the evaluation metrics as defined by Hori et al. [HYKS10].

2.2.4.2 ASIC vs FPGA



(a) Arbiter PUF #1 (b) Arbiter PUF #2 (c) Loop PUF

FIGURE 2.16 – Intra-device evaluation.

In this subsection, we compare the PUF_{mix} intra-device performance when

designed on two different platforms, FPGA and ASIC, using the CMOS 65nm technology. Three performance indicators are investigated in order to evaluate the PUF_{mix} (2 arbiter PUFs and 1 loop PUF) performance on both platforms, which are randomness, uniqueness and steadiness.

To characterize the randomness of each arbiter and each loop PUF, we compute the min-entropy of the bit response sequence when applying the same set of different control words to the 49 PUF instances. Figure 2.16a and Figure 2.16b show the average of intra-device evaluation results of the 49 arbiter PUF #1 and the 49 arbiter PUF #2, respectively on the two used platforms. On the FPGA, the randomness of the arbiter PUF #1 is 0%. This means that, despite checking the routing time at design process, there is a big bias between the two parallel paths due to imperfect routing. The bit response of the PUF is stable (always at '0' or '1') even when changing the control word. The intra-device evaluation of the arbiter PUF #2 shows that the bias on FPGAs is reduced and the randomness of the arbiter PUF #2 increases to 25%. However, on ASIC, the two arbiter PUFs present almost the same performance results. Then, we can conclude that, due to manual routing, the arbiter PUF design on ASIC is slightly better in terms of randomness (around 30%) than on FPGA.

Using the min-entropy of the bit response sequence obtained when introducing the same set of challenge $T = 128$ times, we assess the steadiness of the PUF. Figures 2.16a and 2.16b show the average steadiness of the 49 arbiter PUF #1 and the 49 arbiter PUF #2, respectively. Since there is a bias on the design of the arbiter PUF structures (poor randomness), we cannot judge the steadiness which is around 100% on both platforms. Therefore, the steadiness of the arbiter PUFs cannot be investigated even on the ASIC platform since it is influenced by the low randomness characteristic (around 30%). Our results show (Figure 2.16c) that, on both platforms, the loop PUF presents a good randomness characteristic (around 100%), since there is no need for routing constraints. In this case the steadiness of the PUF can be investigated. And the average steadiness of the 49 loop PUFs is around 98% on ASIC and FPGA platform. This proves that the loop PUF design is very reliable independently of the used platform.

The intra-device uniqueness of the PUFs response is studied using the SHD metric. Although both platforms are built with the CMOS 65nm technology, due to the noise of designed and unused components on the FPGA, the extraction process is better on ASIC. This makes the uniqueness of the designed PUFs (arbiter and loop PUFs) better on ASIC than FPGA.

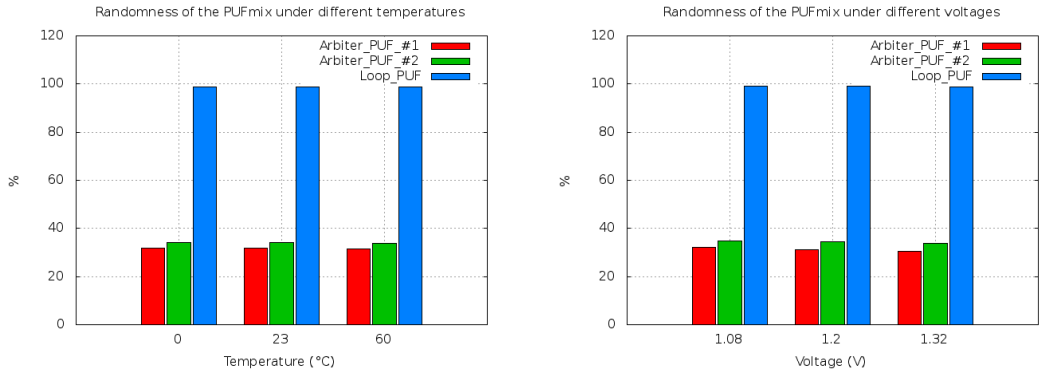
We conclude that the PUF_{mix} design presents better performance on ASIC than FPGA. First, the randomness of the arbiter PUF is better on ASIC than on FPGA. Second, the intra-device uniqueness is higher on ASIC.

In Section 2.2.4.3, we present a deeper analysis of the PUF_{mix} performance under varying environmental conditions and when performing inter-ASICs evaluation.

2.2.4.3 Deeper Analysis on ASIC : arbiter vs loop PUF

In this subsection, we start our investigation by the evaluation of the randomness and the steadiness of the arbiter and the loop PUF structures when varying operating conditions. Then we study the inter-device characteristics of the PUFs when placed in the same places in different ASICs.

Randomness Analysis



(a) Temperature impact (b) Voltage impact

FIGURE 2.17 – PUFmix randomness evaluation

We randomly choose a set of 1024 challenges for the evaluation of the randomness of the arbiter and the loop PUF instances under different operating conditions. As proposed by Hori et al. [HYKS10], we compute the min-entropy of the obtained bit sequence to evaluate the randomness of each PUF when applying the 1024 fixed challenges.

Figure 2.17a presents the mean value of the randomness percentage obtained for the 49 PUFs for each structure when performing under different temperatures. The results show that both arbiter PUFs present poor randomness characteristics. The highest randomness value for the arbiter PUFs is 34.03%. It is obtained when performing under normal operating conditions for the arbiter PUF #2. However, the best randomness value for the loop PUF is obtained under the nominal operating conditions (98.92%). When varying the supply voltage ($\pm$ 10% of the nominal 1.2 V), the evaluation of the randomness of two arbiter PUFs and the loop PUF shows that the randomness of both structures is independent from the supply voltage variation (Figure 2.17a). The loop PUF is far better than the arbiter PUF in terms of randomness even when varying the supply voltage. The higher randomness value for the arbiter PUFs is 34.73%, however, the lower value for the loop PUF is 98.94%.

We conclude that there is a big gap in the randomness performance between the arbiter and the loop PUF structures on the ASIC platform. The loop PUF

is better than the arbiter PUFs when varying the temperature (0°C, 23°C and 60°C) and the supply voltage (± 10% of the nominal 1.2 V).

As discussed in Section 2.2.4.2, the poor randomness of the arbiter PUFs can be explained by the imperfect balance between the two paths. The imperfect balance of paths influences negatively the randomness of the arbiter PUF and positively on the steadiness of the PUF. Indeed a very poor randomness increases the steadiness as the PUF output has more probability to be steady if it as often the same value. Therefore, the evaluation of the steadiness is not appropriate in the case of low values of the randomness (for arbiter PUFs in our case).

Steadiness Analysis

The steadiness, or reliability, is the property that a PUF always generates the same response even when varying operating conditions, supply voltage and temperature. Using the Hori [HYKS10] metrics, we determine the average steadiness of the response of the 49 PUFs designed on aselected ASIC when applying a fixed random challenge. This analysis is done under different ambient temperatures (0°C, 23°C and 60°C) and supply voltages (± 10% of the nominal 1.2 V). As mentioned above, the reliability of the arbiter PUFs is not very relevant since it is very influenced by the bias between the paths (low randomness).

When varying the temperature, the loop PUF steadiness remains stable. This means that loop PUF instances does not depend on the operating temperature. However, the supply voltage variation influences the loop PUF steadiness performance. At nominal supply voltage the loop PUF steadiness is the highest (=98.26%). When varying the operating voltage (± 10% of the nominal 1.2 V), the steadiness of the loop PUF is slightly reduced (around 92.5%). The results illustrated in Figure 2.18a and 2.18b show that the loop PUF instances have a good steadiness (higher than 92%), which can be handled using a lightweight error correction schemes.

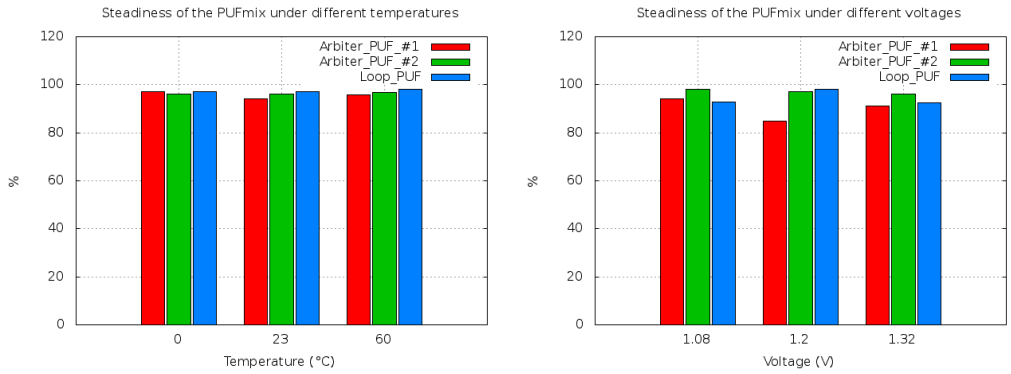
We conclude that the loop PUF and the arbiter PUF instances have a good level of steadiness against variation of temperature and supply voltage, but the arbiter PUF takes advantage of its low randomness.

Inter-Uniqueness Evaluation

The inter-uniqueness shows the ability of a PUF to produce different responses when designed in the same place in identical devices when applying the same challenge under normal operating condition. We use the Hori method, which is based on the computation of the SHD of equivalent bit response.

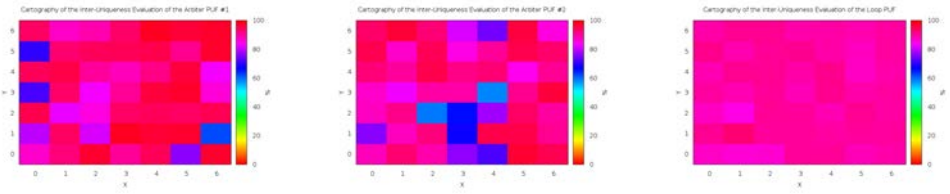
Figures 2.19a, 2.19b and 2.19c present a cartography of the inter-uniqueness evaluation over the 18 ASICs of the arbiter PUF #1, the arbiter PUF #2 and the loop PUF, respectively.

Our results show that the loop PUF has the best average inter-uniqueness characteristics. Almost all the 49 loop PUFs have an inter-uniqueness around



(a) Temperature impact (b) Voltage impact

FIGURE 2.18 – Loop PUF steadiness evaluation.



(a) Arbiter PUF #1 (b) Arbiter PUF #2 (c) loop PUF

FIGURE 2.19 – Inter-uniqueness evaluation.

90%. However, the variance of the inter-uniqueness of the 49 arbiter PUFs #1 and the 49 arbiter PUFs #2 is very large. The PUF located in cell (1,0) has the least inter-uniqueness value of 86.08%. However, both arbiter PUFs present lower inter-uniqueness characteristics than the loop PUF. The lowest values are 61.83% and 57.90% for the arbiter PUF #1 (located in cell (6,1)) and the arbiter PUF #2 (located in cell (4,3)), respectively.

We conclude that the inter-uniqueness of the loop PUF is better than the inter-uniqueness of both arbiter PUFs. Regardless of its location on the ASIC, the loop PUF structure presents similar inter-uniqueness performance.

2.3 Conclusions

In this chapter, we presented a novel structure of silicon delay PUFs. It is referred to as "loop PUF". It is based on N identical and controllable delay chains which are serially assembled in a loop to create a single ring oscillator. It offers a huge number of challenges.

After that, we proposed a detailed discussion on the implementation steps of a delay PUF design referred to as "PUF_{mix}" including two types of delay PUFs

on different platforms using the same 65nm technology. We have implemented the design on both ASIC and FPGA. The number of 168 instances was implemented on the FPGA and we have developed and produced an ASIC carrying 49 PUF_{mix} designs. We start comparing the performance of the PUF design on FPGA to its performance on ASIC. We concluded that according to the uniqueness experimental results, the PUF_{mix} presented better performance results on ASIC that on the Virtex-V FPGAs. Then we performed a deeper analysis of the PUF_{mix} performance when designed on ASICs. Each PUF_{mix} structure is composed of two arbiter PUFs and one loop PUF. A large scale performance analysis, concerning their randomness, uniqueness and reliability, is performed on the 18 manufactured ASICs under different operating conditions. We concluded that the loop PUF presents better inter-uniqueness performance. However, we cannot compare their steadiness performance due to the bias of the arbiter PUF structure explained by its low randomness performance.

In the next chapter, we present another novel PUF structure based on the transient effect ring oscillators (**TERO**) principle. We propose a detailed discussion on its implementation on FPGA structures. Then, we present its performance evaluation.

Chapitre 3

TERO PUF

This chapter focuses on the presentation of a novel PUF structure, its implementation on FPGA platforms and the evaluation of its performance. The proposed structure is referred to as "TERO PUF". It is a ring oscillator structure which can also be used as a TRNG, thus providing a significant advantage compared to other PUFs. The PUF response takes advantage from the introduced oscillatory metastability of an SR flip-flop when their S and R inputs are connected to the same input signal. We detail the implementation steps required for the validation of the "TERO PUF" structure. We list indeed the problems that can occur and the applied solutions. Then we evaluate the structure when implemented on multiple ALTERA Cyclone-II FPGAs. Each FPGA embed four(4) 64-loops TERO PUFs. The work presented in this chapter is the result of a joint work with X. Ngo [Ngo12].

3.1 TERO PUF

In this section, we propose a new PUF structure that exploits the oscillatory metastability of cross-coupled elements. This PUF is based on transient effect ring oscillator (TERO) cells and referred to as "TERO PUF". The main benefit of this structure is that it is not sensitive to locking phenomenon. Indeed, unlike RO-PUFs, the oscillation frequency is not taken into account. The proposed PUF uses the number of oscillations as an entropy extractor that does not depend on the locking phenomenon.

The TERO cells are originally proposed by Varchola et al. [VD10] for TRNG designs. Hereafter, we present the TERO cell structure and study how it can be used to extract the process variation entropy to generate a PUF response.

3.1.1 TERO loop Architecture and Behavior

The basic element of the TERO PUF structure is the TERO loop. It is a bi-stable circuit. It is composed of an SR flip-flop. When connecting the S and R inputs of the flip-flop to the control input signal *ctrl* we generate a metastability of oscillations. The structure proposed by Varchola et al. [VD10] uses XOR and AND gates to control the loop. In order to simplify the control mechanism of the TERO PUF, we use AND gates and inverters to stimulate the oscillatory metastable state. Usually two inverters are used, but we can add as much as we want to extend the oscillation period of the loop. Figure 3.1 shows the TERO loop architecture.

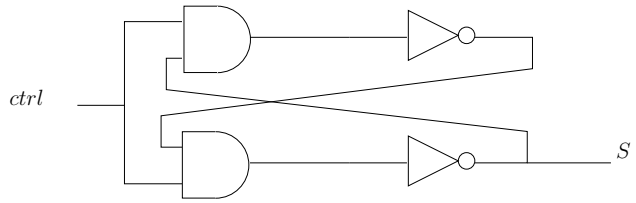


FIGURE 3.1 – Architecture of a TERO loop.

We denote by *ctrl* the edge input signal that triggers the TERO loop structure to stimulate it. This signal causes transitory oscillations in the loop if the following two conditions are fulfilled [VD10] :

1. The circuit must have a positive feedback.
2. The RC time constant must be shorter than the total delay of the logic elements used in the loop. The RC time is defined by the parasitic resistance R and the capacity C.

In theory, if the loop is ideally symmetrical, oscillations will never stop. However, the loop only oscillates for a short time after which the oscillation stops due

to intrinsic asymmetry. Varchola et al. [VD10] denote this intrinsic asymmetry by a factor Td as the time difference between the time delays of both halves of the loop. The authors claim that the number of oscillations is proportional to the value of Td. They also claim that the average Td value of a well balanced TERO loop, is related to random contributions to gate delays, which originate in semiconductor intrinsic noise. In this way, the number of oscillations is significantly affected by the random circuit noise. Thus, the number of oscillations varies at the end of successive simulation intervals. On the other hand, global perturbations (*e.g* from the power supply) do not significantly affect the Td parameter because of the $differential$ behavior of the loop.

The TERO loop looks like a bi-stable butterfly element (which does not oscillate) as proposed by Kumar et al. [KGM+08]. Both of their outputs fall to '0' or '1' as a final state depending on the manufacturing process variation. The only difference is that, due to the applied *ctrl* input signal, the TERO loop has a longer delay causing a metastable oscillatory behavior as explained above in this section. Figure 3.2 shows an electrical behavior of two different TERO loops. The *ctrl* input signal is forced to '1' for 2.5μs its frequency is 200 KHz) on both loop elements. As illustrated in Figure 3.2, the number of oscillations in the two TERO loops outputs $S_{\#1}$ and $S_{\#2}$ is not the same. Moreover, the final logic levels can also be different as observed in Figure 3.2.

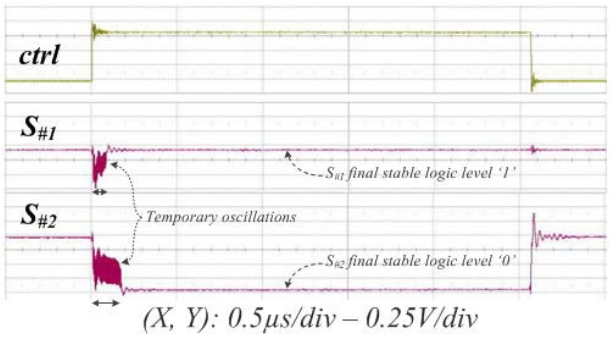


FIGURE 3.2 – Electrical behavior of two TERO loops.

Either, the number of oscillations or the final logic state of the loop can be used as a source of entropy to generate a PUF response. Both of them depend only on the randomness variation during the manufacturing process. The *ctrl* input signal can be used as a PUF challenge. Hereafter, we study which method is better to use to extract this process variation entropy.

3.1.2 Extraction of the Process Variation Entropy

We first test the final logic state of the output signal of the loop. We place the TERO loop in nine FPGAs. Each one contains 1172 TERO loops. For all

of the 10 548 TERO loops tested, we obtain the following results : in 40.3% of the cases, the output signal falls into the stable logic state '0', in 30.7% of the cases, the output signal falls into the stable logic state '1', and in 29% of the cases the output signal is unstable (oscillating). This unstable state is a major drawback when we use the final logic state as a PUF response. Even when we extend the period of the *ctrl* signal, we obtain similar results. We note that for about a third of tested TERO loops, the Td factor is quite small. This can be explained by the random symmetry of the process variation. Similar results are reported by Yamamoto et al. [YSI+11]. Unlike [YSI+11], we conclude that the final state of the output signal cannot be used as a source of entropy to generate the PUF response.

Next, we tried to use the number of oscillations of the output signal S as a source of entropy. We use an 8-bits counter to measure the number of the oscillations of each TERO loop. For all the 10 548 TERO loops, we measure 2^{18} times the number of oscillations. In all the experiments, temporarily oscillating signals do not oscillate more than 255 periods and an 8-bit counter is hence sufficient.

For each of the temporarily unstable TERO loops, the number of oscillations is normally distributed. According to the central limit theorem, the distribution of the number of oscillations is due to the sum of many independent variables. Clearly, the mean value of this distribution is related to the characteristics of the chip which is determined by the manufacturing process. Then, the mean value can be used as a source of entropy to generate the PUF response. However, the standard deviation of the distribution depends on the electrical noise of the circuit. According to our experimental results for the temporarily unstable TERO loops, due to the electrical noise introduced on the circuit the mean number of the four least significant bits (LSBs) of the 8-counter are not stable from a test to another. Thus, they cannot be used as a PUF response. But they could be exploited as a source of randomness for TRNGs. However, the mean values of the most significant bits (MSBs) are stable form a test to another but they vary significantly from a TERO loop to another. For the temporarily oscillating loops, the same MSBs can be used as a PUF response.

Therefore, we propose for both stable and unstable TERO loops to use the mean number of oscillations as a source of entropy to generate the PUF response. Hereafter, we detail the proposed TERO PUF structure.

3.1.3 TERO PUF structure

The proposed TERO PUF uses the number of oscillations of the TERO loop as a PUF response. Based on previous experiments, a binary asynchronous 8-bit counter is needed to measure the oscillation frequency of the TERO loop element. We suggest also taking into account only the mean value of the number

of oscillations to generate the TERO PUF response in order to enhance the steadiness of the generated value. Therefore, the TERO PUF structure includes a 26-bit accumulator and an 18-bit shift register.

Generally, differential structures are recommended for the PUF structures. They reduce the noise perturbation in electronics systems. However, differential structures are area costly in our case, especially if we need a large TERO PUF response. Figure 3.3 shows the differential structure of the proposed TERO PUF which takes this constraint into account. Its purpose is the subtraction between the 8-bits responses (average number oscillation) of adjacent TERO loops. We considered only at most the four MSBs of the subtraction results in order to eliminate the instabilities caused by the electrical noise.

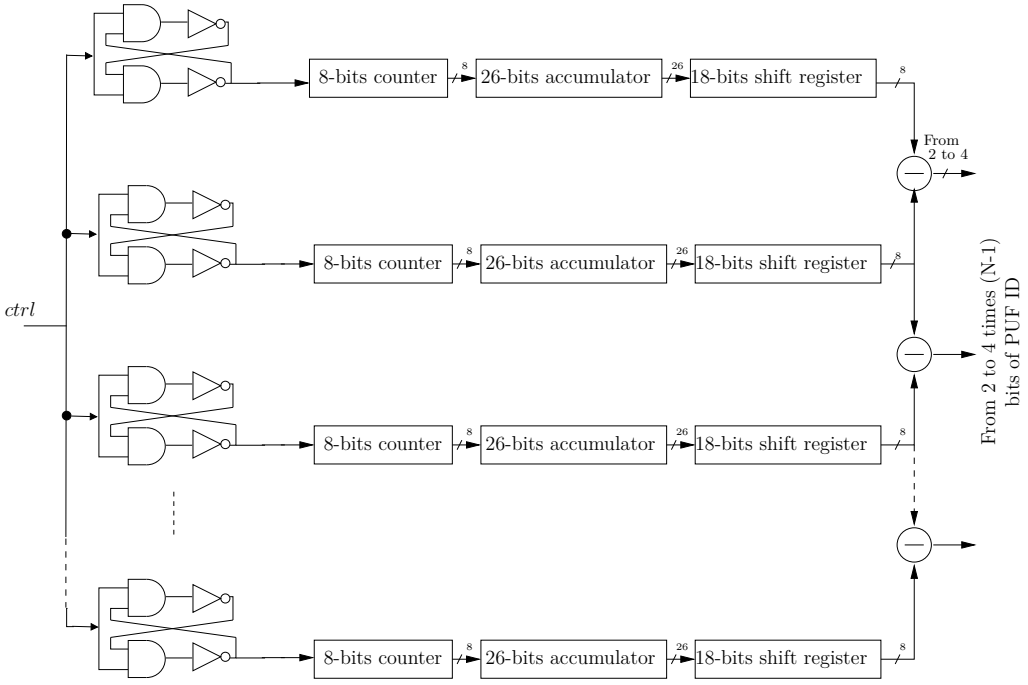


FIGURE 3.3 – TERO PUF architecture.

Hereafter, we propose to evaluate the differential TERO PUF structure after implementing it into ALTER FPGAs.

3.2 TERO PUF on ALTERA FPGAs : Implementation and Evaluation Results

In this section, we propose to evaluate the performance of the TERO PUF structure. We implement the TERO PUF design in different places in a same FPGA in order to evaluate its performance according to its placement. The same

structure is also implemented in different FPGAs to provide an inter-device performance characterization of the proposed TERO PUF.

3.2.1 TERO PUF IP : Design Requirements

The TERO PUF Module

We propose to implement a TERO PUF with 64 TERO loops. Using a 64 TERO loops, the 64-loops TERO PUF can generate a response of 252 bits when only the four MSBs of the shift register are considered for the subtraction phase. We implement four instances of the 64-loops TERO PUF on the same FPGA platform. To be sure that the extracted PUF response depends only on the manufacturing variability, ideally, we have to manually route and place the 64 TERO loops. Since on FPGAs we are obliged to use pre-placed and routed cells, hard placement constraints are needed to build the TERO loops.

As discussed on Section 3.1.2, we consider the mean value of the number of oscillations of the TERO loops to generate The PUF response. For each TERO loop we use a succession of counter, accumulator and shift register to compute the average number of its oscillation (See Figure 3.3).

The Control Part

We implement four 64-loops TERO PUFs instances on each FPGA. The selection and the activation of the desired 64-loops TERO PUF structure is carried out by the defined Finite State Machine (FSM). Only one 64-loops TERO-PUF is activated at one time. The control part interact with an SRAM to save all responses of each 64-loops TERO PUF in order to analyse them afterward. It also manages the counter values.

The Interface Part

All the control commands that we need to activate or to save the TERO PUF response pass through the FSM. However, the rising edge of the *ctrl* signal used to stimulate the TERO loops is directly connected to the TERO PUF modules. To analyse the PUFs responses, we propose to use an output signal to extract the saved data from the SRAM block. Table 3.1 shows all needed interface signals. Figure 3.4 indicates the top-level architecture of the TERO PUF IP.

TABLE 3.1 – TERO PUF communication interface (I/0).

`FSM_control`	in	(2 bits) Input data.
`Clock`	in	Clock system.
`Ctrl`	in	(1bit) Loops stimulation signal.
`Data_out`	out	(8bits) Output data.

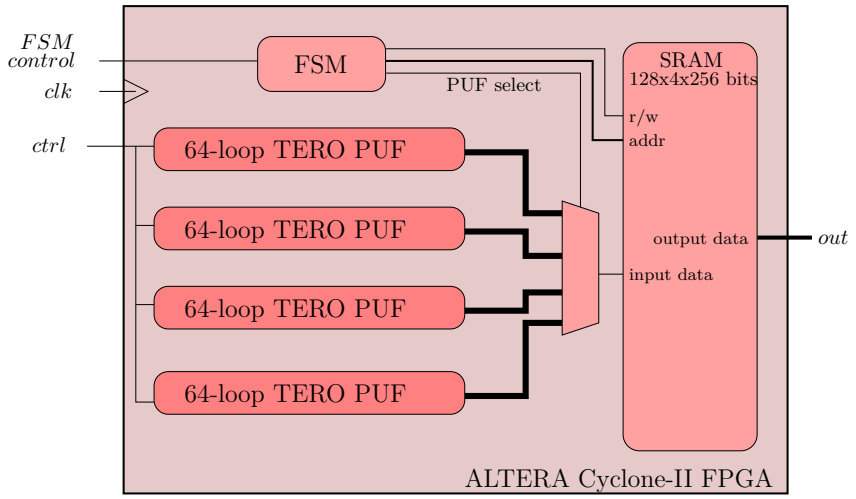


FIGURE 3.4 – Top-level architecture of the PUF IP.

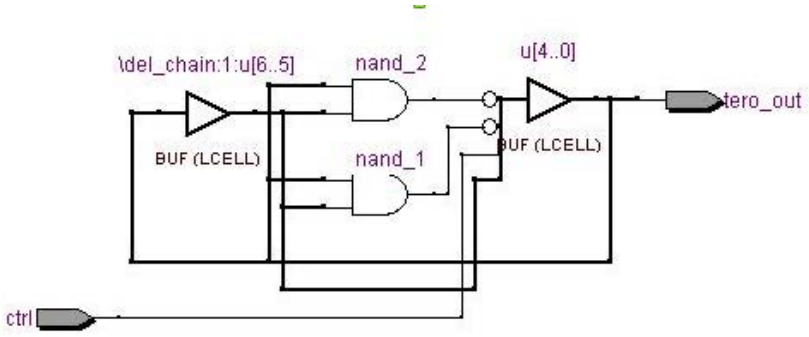
3.2.2 FPGA Implementation Details

We designed four 64-loops TERO PUFs on nine ALTERA DE1 boards featuring Cyclone-II FPGA. The TERO loop is the main logic element from which we extract the process variation entropy. Specific compilation, routing and placement efforts are needed to implement them in order to guaranty that the PUF response takes advantage only from the manufacturing process variations. The counters, the accumulators and the shift registers are synthesised automatically by the ALTERA software "Quartus". Also, to fairly compare the number of oscillations of two TERO loops, we must ensure that all designed TERO loops are identical either in the same TERO PUF or not. Hereafter, we discuss two problems that can occur when implementing the TERO loops, the simplification and the place and route problems.

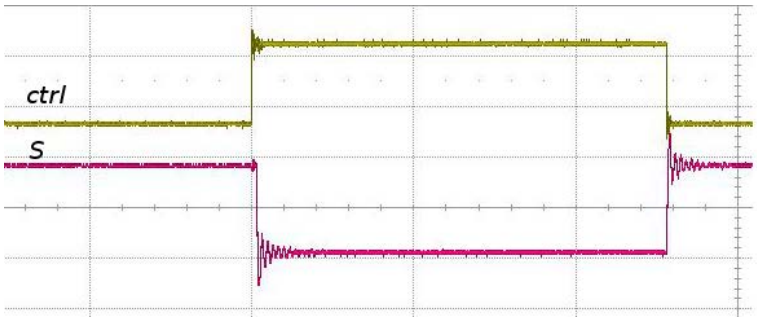
3.2.2.1 Design Simplification Problem

We first describe our design using the VHDL language. When we compile it, the Quartus software applies some simplifications and produces a simpler design which offers similar results. In our case Figure 3.5a shows that the compiler designs two NAND gates instead of two ANDs and two inverters. This simplification completely changes the operating purpose of our TERO loop. Figure 3.5b shows that the output signal does not oscillate when we activate the control signal *ctrl*. However, we use the number of oscillations before reaching the stable state to generate the PUF response. We have ; therefore, to add a constraint to keep all defined internal signals for the TERO loop at the design stage to produce desired oscillations. Figure 3.6a and 3.6b show the obtained design and the corresponding electrical behavior when we add the "**keep**" attribute for all internal signals,

respectively. We can now approve our TERO loop design model.



(a) Schematic.



(b) Electrical behavior.

FIGURE 3.5 – TERO loop simplified design.

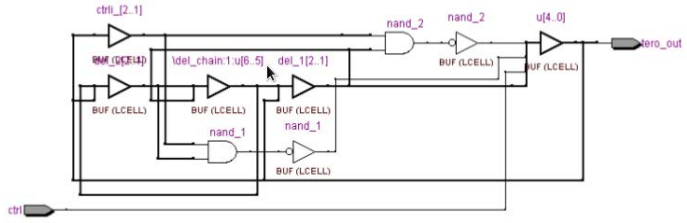
3.2.2.2 Place and Route Problems

Before introducing the problems that occur when designing the TERO loop module, we first describe the ALTERA Cyclone-II FPGAs architecture. The latter includes different blocks (Figure 3.7) :

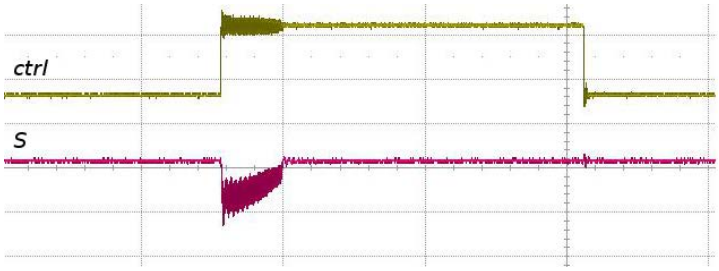
- Phase-locked loop (PLL) blocks ;
- Input/Output blocks ;
- Internal memory blocks referred to as M4K blocks ;
- Embedded multipliers block ;
- Logic array blocks.

The Logic Array Blocs referred to as “LABs” are the most important ones. They embed the elementary cells (Logical Elements (LEs)) used to implement our design. All LABs are similar, but the logical elements inside the LABs are different. The ALTERA Cyclone-II FPGA that we use contains 1127 LABs. Each one includes 16 LEs as shown on Figure 3.8.

The TERO loop implementation needs 15 LEs corresponding to the 15 logical gates used in the TERO PUF design (Figure 3.6a). Figures 3.9a and 3.9b show



(a) Schematic.



(b) Electrical behavior.

FIGURE 3.6 – TERO loop desired design.

that the number of oscillations of the TERO loop is highly dependent on its placement on the same LAB. It depends on which LEs we use. When we use the first two LE and the last thirteen ones on a LAB, the output of the TERO loop oscillates more than when we use the first seven LEs and the 8 last LEs in the same LAB. In order to fairly compare the TERO loops, we must use the same LEs located in different LABs for all designed 64-loops TERO PUFs. Also, in order to evaluate the intra-device properties of the TERO PUFs, we reproduce the same design TERO PUF four times in the same FPGA using the same LEs for all TERO loops. On ALTERA Cyclone-II FPGAs, it is very simple to reproduce the same design. We start first by designing a reference TERO loop element. Then we copy and past it 64 times to design a reference 64-loops TERO PUF. The latter is then copied four times into the same FPGA to perform intra-device evaluations.

The basic element that constitutes a LE is a Look Up Table referred to as "LUT" (Figure 3.10). The LUT is used to create the basic combinatorial functions of our design. It has four data inputs (data1, data2, data3 and data4) and one output signal. Figure 3.10 shows that there are two input types for each LUT three direct inputs (data1, data2 and data4) and one indirect data input (data3).

At the compilation phase of the design, the compiler randomly chooses an input path. However, in our case, if two TERO loops use different input paths, the output response of the PUF will be biased. Figure 3.11a shows a TERO loop electrical behavior when using 14 LEs with direct links and only one LE using

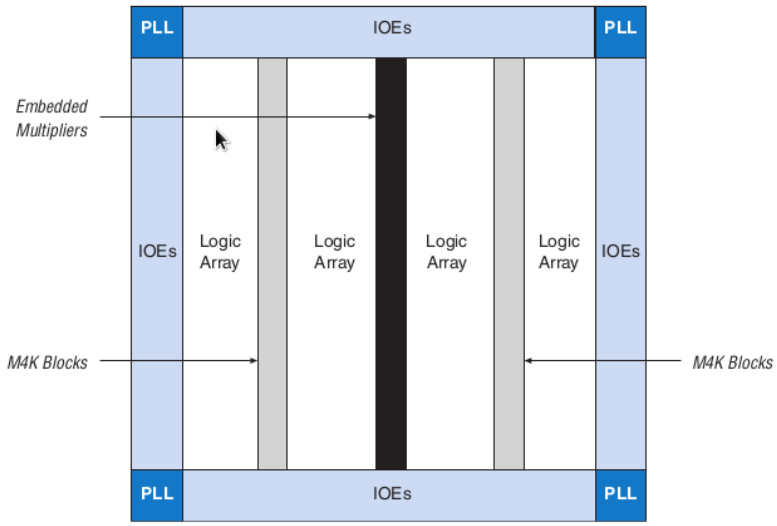


FIGURE 3.7 – Cyclone-II block diagram [ALT].

an indirect link. Figure 3.11b shows a TERO loop electrical behavior when using only direct links. We note that in the first case (one LE uses indirect link) the TERO loop oscillates less than when we use only the direct links. This can be explained by the delay added by the multiplexer used for the indirect link of the LUT. In our case, to be sure that we take advantage only from the process variation of two TERO loops, we have to ensure that on our design we use the same LUT entries for all equivalent LEs of the TERO loops.

We conclude that to deal with the simplification and the placing and routing problems, we have to :

1. add the "keep" attribute to the internal signals of the TERO loop design.
2. define placement constraints to use equivalent LEs for different TERO loops in different LABs.
3. specify which LUT entries have to be used on each LE. In our case, only direct links of the LUTs are used.

3.2.3 Metrics Definition and Experimental results

As discussed in Section 3.1.2, the TERO loop should exploit the mean number of oscillations of the output signal of the TERO loop. Our experimental results show that only MSBs of the mean number of oscillations have to be considered to generate the PUF responses. Table 3.2 presents the average number for all statistical results (randomness and steadiness) of the 8-bits output of the subtraction phase with the following experimental setup : 64 TERO loops, 63 subtractions, 4 placements, 9 boards and 128 samples by measurement. According to the metrics that we use (see Section 3.2.3.1), the best randomness is obtained when we

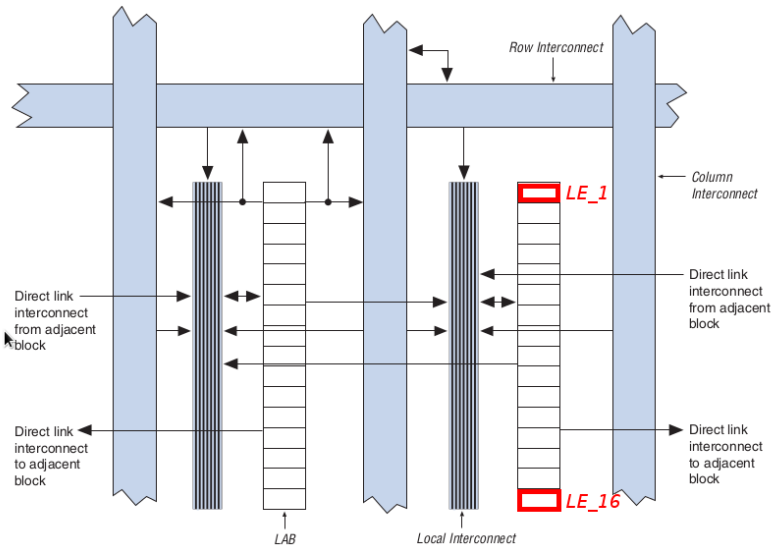


FIGURE 3.8 – Cyclone-II Logic Array Blocks [ALT].

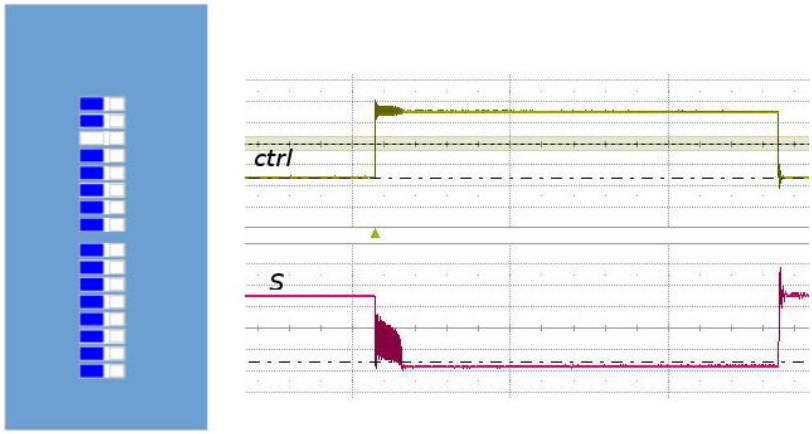
have 50% of logic level at 50% and the best steadiness corresponds ideally to 0%. According to the obtained results, the two MSBs (Bit#7 and bit#6) of the shift register can be used as a PUF response even in critical application such as cryptographic key generation purposes since they show a very low steadiness (less than 2%). Bit#3 and Bit#4 can be used fore some applications which are not critical such that device authentication. They still give acceptable results. However, the four LSBs are not stable at all. They show a steadiness between 9.1% and 39.5%. With these results they could not be used as a PUF response even in non critical applications. As a consequence, the TERO architecture is fully scalable ; using 64 TERO loops, the TERO PUF can generates a response of 126 bits, 189 bits or 252 bits, when using 2, 3 or 4 bits of the difference, respectively. Hereafter, we first define the metrics that we use, then we present the evaluation results of the 64-loops TERO PUF with an ID size of 126,189 and 252 bits.

3.2.3.1 PUF Characterization Metrics

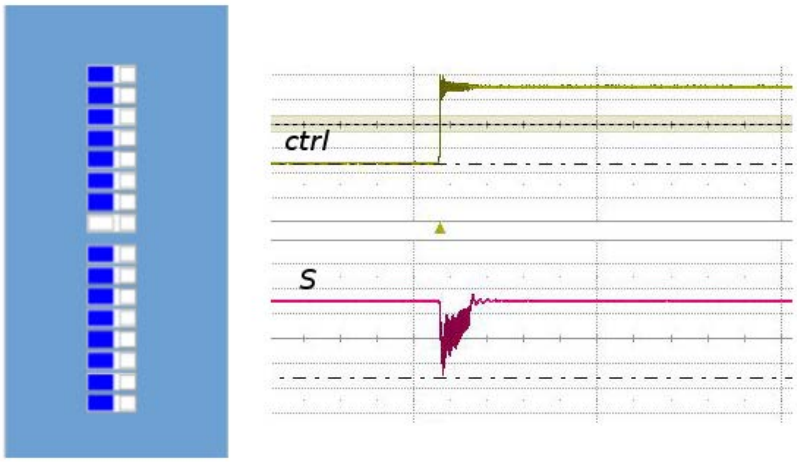
As discussed in chapter 1, there are three main performance indicators to evaluate the PUF performance : randomness, uniqueness and steadiness.

The Randomness also referred to as bias. It is evaluated by measuring the bit bias of the TERO PUF IDs. The evaluation results are given on Table 3.2. Ideally the PUF response contains as much '0' as '1'. We have then 50% of logic level at '0' in the case of best randomness.

The steadiness or intra-die variation quantifies changes of the PUF output over several measurements. Since we don't have enough chips at our disposal for well founded statistical tests, we consider four PUF implementations in the



(a) Placement case#1.



(b) Placement case#2.

FIGURE 3.9 – TERO loop electrical behaviours.

same FPGA as four realizations of the same PUF on different devices. An L-bits reference response $\bar{R}_n^{Pl}$ is first estimated from a chip n and placement Pl in normal operating conditions. Then we compare it to T performed measurements using the HD method in order to evaluate the steadiness of the PUF placed at Pl on the chip n. This can be expressed by

$$\mathsf{Stead}_{\mathsf{n}}^{\mathsf{Pl}} = \frac{1}{\mathsf{T}} \sum_{\mathsf{t}=1}^{\mathsf{T}} \frac{\mathsf{HD}(\bar{\mathsf{R}}_{\mathsf{n}}^{\mathsf{Pl}}, \mathsf{R}_{\mathsf{n,t}}^{\mathsf{Pl}})}{\mathsf{L}} \times 100\%. \tag{3.1}$$

A lower intra-die HD (close to 0%) results in a more stable PUF response.

The uniqueness or inter-die variation shows how match a PUF response is unique even when placed in the same place on different devices. We consider the average between-die HD of the PUF as an estimate of the uniqueness property

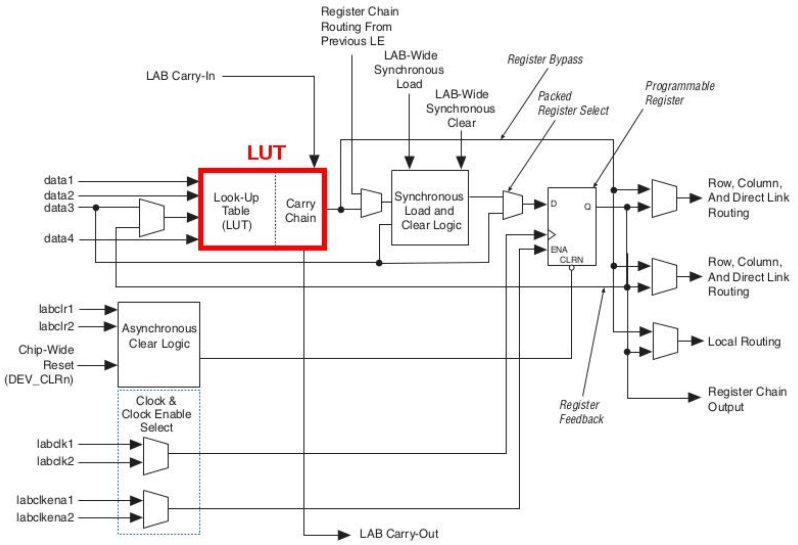
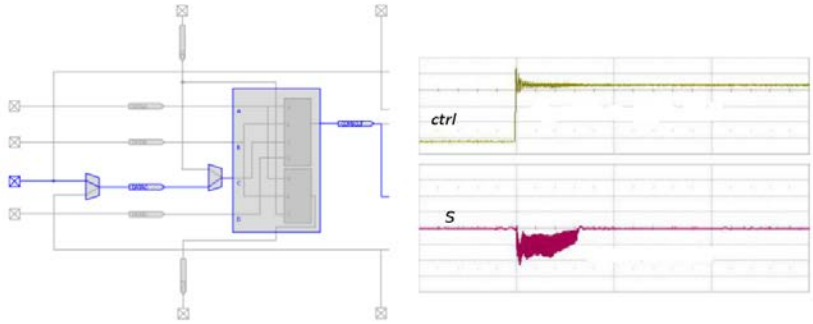
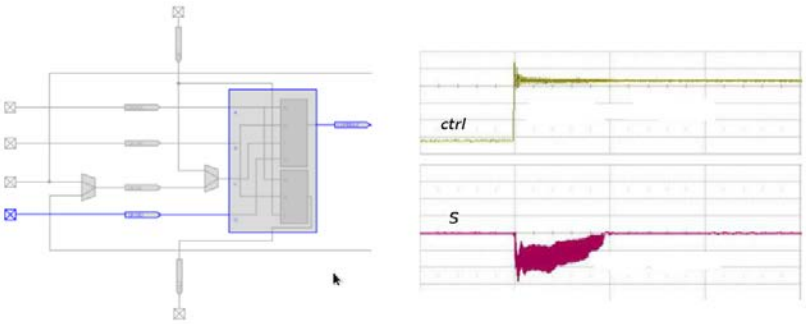


FIGURE 3.10 – Cyclone-II Logical Element [ALT].



(a) Using one indirect input of a LUT.



(b) Using only direct inputs of a LUT.

FIGURE 3.11 – TERO loop electrical behavior.

of a PUF when placed at Pl. The average uniqueness of the PUF response when placed at Pl for the set of N chips is defined as

TABLE 3.2 – Evaluation of the bias and the steadiness of subtraction 8-bit outputs for TERO PUF

	% of logic level '0'	Steadiness (%)	Used in TERO PUF response
Bit#7	54.2	1.7	Yes
Bit#6	54.2	1.7	Yes
Bit#5	52.3	2.7	Yes
Bit#4	51.7	4.8	Yes
Bit#3	51.6	9.1	No
Bit#2	51.8	17.2	No
Bit#1	52.7	30.1	No
Bit#0	50.3	39.5	No

$$\mathsf{Uniq}^{\mathsf{Pl}} = \frac{1}{\mathsf{N}(\mathsf{N}-1)\mathsf{T}} \sum_{\mathsf{u}=1}^{\mathsf{N}} \sum_{\mathsf{v}=1;\mathsf{v}\neq\mathsf{u}}^{\mathsf{N}} \sum_{\mathsf{t}=1}^{\mathsf{T}} \frac{\mathsf{HD}(\bar{\mathsf{R}}_{\mathsf{v}}^{\mathsf{Pl}}, \mathsf{R}_{\mathsf{u,t}}^{\mathsf{Pl}})}{\mathsf{L}} \times 100\%. \tag{3.2}$$

This metric includes all possible pairwise comparisons among N distinct chips tested. For a truly unique response of a PUF, the uniqueness should be close to 50%.

3.2.3.2 Experimental Results

Each 64-loops TERO PUF is tested for four placements ($Pl = 4$) in nine Cyclone-II FPGAs ($N = 9$) and 128 tests are performed for each PUF ($T = 128$). All experiments are done at nominal operating conditions ($T = 28°C$ and nominal $V_{core} = 1.5V$). The frequency of the *ctrl* signal is 5O MHz. Hereafter, we present the performance results of the 64-loops TERO PUF for the different ID size 126,189 and 256 bits.

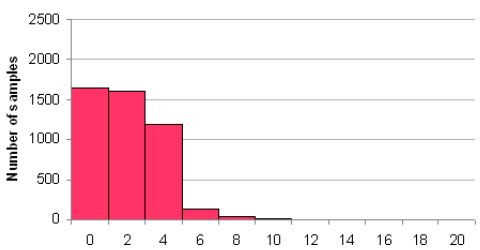
- The mean value of the steadiness of the TERO PUF with 126 bits (we consider only the two MSBs of the 8-bits subtraction result) is 1.73%. This means that the intra-die HD is less than 2.2 bits among 126 bits.
- The mean value of the steadiness of the TERO PUF with 189 bits (we consider only the three MSBs of the 8-bits subtraction result) is 2.07%. This means that the intra-die HD is less than 3.9 bits among 189 bits.
- The mean value of the steadiness of the TERO PUF with 252 bits (we consider only the two MSBs of the 8-bits subtraction result) is 2.75%. This means that the intra-die HD is less than 7.0 bits among 252 bits.

Figures 3.12a, 3.12b and 3.12c shows the histograms of TERO PUF intra-die variation obtained experimentally with an ID size of 126, 189 and 252 bits at nominal operating conditions. The results show the good stability of the TERO

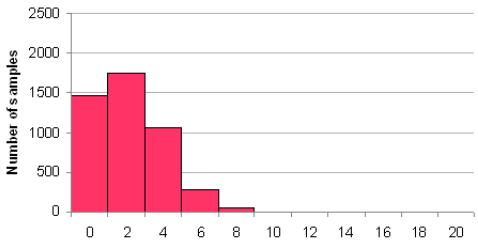
TABLE 3.3 – Characteristics of the 64-loops TERO PUF

PUF ID size (bits)	Intra-die variation(%)	Inter-die variation (%)	Number of used LABS in Cyclone-II
126	1.73	48.07	416
189	2.07	48.99	416
252	2.75	49.27	416

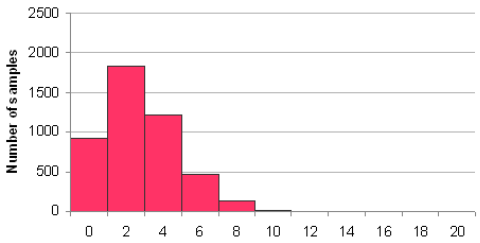
PUF response. Nevertheless, it is clear that an additional post-processing stage have to be added (with a lightweight error correcting algorithm) if the required error probability is null.



(a) 126-bits TERO PUF.



(b) 189-bits TERO PUF.

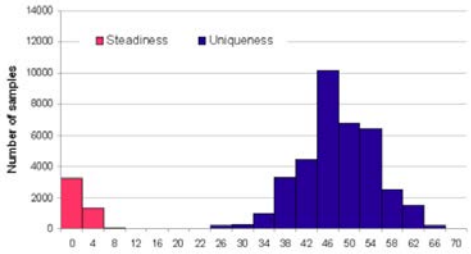


(c) 252-bits TERO PUF.

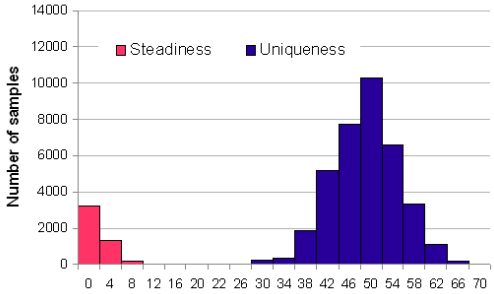
FIGURE 3.12 – Steadiness under normal environmental conditions of

The mean value of the uniqueness of the TERO PUF response is 48.07%, 48.99% and 49.27% for 126, 189 and 252 bits respectively. Figure3.13a, 3.13b and 3.13c shows the histograms of TERO PUF intra and inter die variations obtained experimentally with an ID size of 126, 189 and 252 bits. For each case the two histograms are clearly separated.

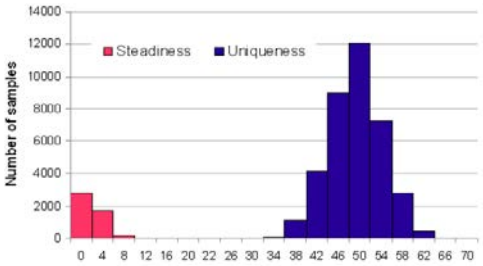
The histograms show that the bigger is the ID size, the better is the PUF uniqueness. Table 3.3 gives a summary of the 64-loops TERO PUF characterization results when implemented on the Cyclone-II FPGAs.



(a) 126-bits TERO PUF.



(b) 189-bits TERO PUF.



(c) 252-bits TERO PUF.

FIGURE 3.13 – Steadiness and uniqueness of

We also studied the steadiness of the TERO PUF structure when varying the temperature from (22°C to 80°C) using an incubator. Figure 3.14 shows that our 64-loops TERO PUF propose better performances at low temperatures(22°C). When we use two or three bits of the accumulator response the PUF shows similar performance. However, when we use four bits of the accumulator response in order to generate a 252 bits response under higher temperatures, the PUF response became very instable. We conclude that the high temperature influences negatively the TERO PUF steadiness performance and especially with an ID size of 252 bits.

3.3 Conclusion

In this chapter, we proposed a novel PUF structure which is based on transient effect ring oscillator (TERO). It is called "TERO PUF". It uses the mean number of TERO loop oscillations as a source of entropy to generate a PUF response which makes it, theoretically, non sensitive to the locking phenomenon.

Then, we presented first the problems that can occur when implementing it in the ALTERA FPGAs. We suggested then a solution for each problem in order to guaranty that we exploit only manufacturing variations to generate the PUF response. Second, we evaluated the performance of the TERO PUF design when

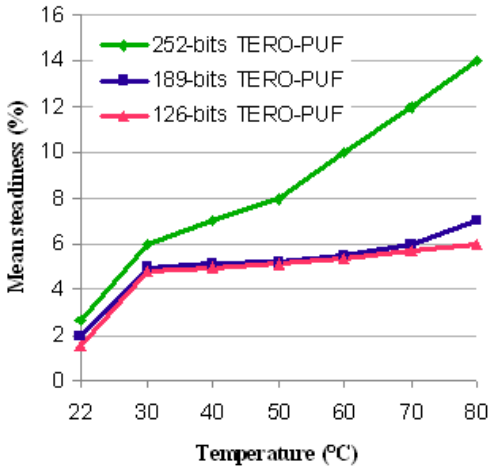


FIGURE 3.14 – Steadiness of the TERO PUF under different temperatures.

implemented four times in nine identical Cyclone-II FPGAs. We concluded that the TERO PUF propose interesting uniqueness and steadiness performance. As a consequence, the TERO PUF is a new candidate for FPGA-dedicated PUF.

In the next chapter, we will focus on the evaluation methods of PUFs structure. We propose new metrics dedicated to the delay PUFs evaluation. It is based on delay statistics which can be performed at design stage of the PUF structures.

.

Chapitre 4

Delay PUF Performance Evaluation Method

The purpose of this chapter is to present a new characterization method which aims firstly at better delay PUFs assessment, and secondly at evaluating the PUF at design stage without the need for a real silicon. With the permanent growth of novel PUF architectures, the need for such methods becomes important. All existing methods in the literature are based on statistical tests of the PUF bit-response. In this chapter we propose a novel PUF characterization method that is specific to delay based PUFs and uses statistical measurements on delay elements. The advantage of this method is to allow the designer to be sure that the PUF has good performances before its effective implementation. Hereafter, we provide details about our method, presenting new metrics and evaluation results on different platforms, for both arbiter and "loop PUF " structures.

4.1 Motivation

As seen in Chapter 1, all proposed methods in the literature use statistical tests on the set of logical PUF IDs to evaluate the PUF performances. They need a considerable number of trials in order to run a Monte-Carlo estimation method. However, our method takes advantage from measurements of the physical values (i.e., the delays or frequencies). It then requires less tries to evaluate the PUF performance than classical methods. The metrics studied to characterize the PUFs are randomness, uniqueness and steadiness. They take advantage from the measured physical values of elementary component making up the PUF. The delay distributions provide the interest to quantify the PUF at the physical level rather than carrying out a lot of experiments to get the PUF ID at logical level as suggested by the previous proposed metrics. The number of tests is indeed linear in M, where M is the number of elements composing the PUF. Moreover, when we perform electrical simulation of delays of the basic delay elements, this method can be applied at design stage. A designer can therefore evaluate the PUF performance before implementing it. Although the benefit of our method, it has two disadvantages. First, the characterization on a PUFs can be realized only by the designer or someone how knows the design and have access to it. Second, this method can be applied only for silicon delay based PUFs. For each delay PUF structure we have to redefine our metrics. The latter are highly dependent from both : the PUF structure and the relation between the PUF response and the delay elements.

Hereafter we first give a comprehensive presentation of our proposed methods and the metrics definition for both arbiter and loop PUFs as defined in Section 2.2.2. We discuss then their advantages and drawbacks. And finally we present experimental results of the arbiter and the loop PUFs when characterized using both, our and Hori metrics.

4.2 Background on Gaussian Probability Density Function

For the evaluation of PUFs using statistical delay measurements, we define three metrics. They are based on computing the error function on the probability density function (*pdf*) of Gaussian distributions. Therefore, all proposed metrics need to know the probability to measure a value below a certain threshold t. Figure 4.1 shows a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ where μ is the mean and σ^2 is the variance. The filled blue area contains values of the Gaussian distribution under a threshold t. Assume that $x \in \mathcal{N}(\mu, \sigma^2)$, then the probability to obtain x such that $x < t$ is given by

$$\Pr(x < t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{t} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx = \frac{1}{2}(1 + \mathrm{erf}(\frac{(t-\mu)}{\sigma\sqrt{2}}). \tag{4.1}$$

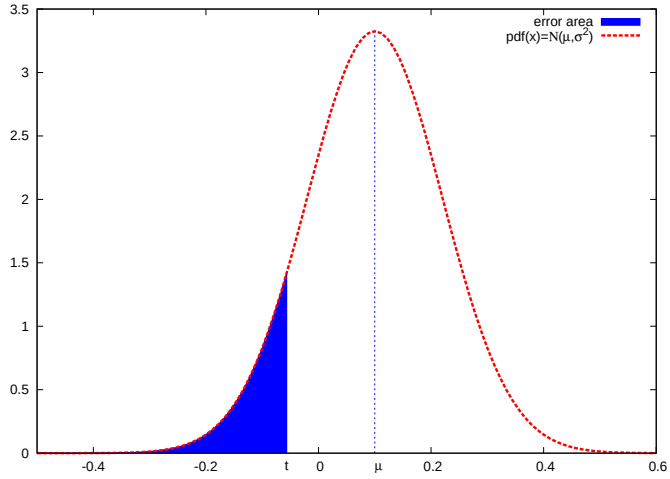


FIGURE 4.1 – Error function and $pdf(x)$.

In mathematics, the error function is a special function (non-elementary). It is defined as

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_{0}^{x} e^{-t^2} dt.$$

The integral in the expression of the error function cannot be evaluated in closed form in terms of elementary functions. However, it can be evaluated by expanding the integral into its Taylor series and integrating all terms one by one. The error function's Taylor series is given by

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}(x - \frac{x^3}{3} + \frac{x^5}{10} - \frac{x^7}{42} + \frac{x^9}{216} - \ ...).$$

The principles explained above in this section are used to define our proposed metrics.

4.3 Proposed Metrics for Delay Based PUFs

We only consider the three well known performance indicators to evaluate PUFs performance which are randomness, uniqueness and steadiness. In this section, we first give a reminder of each performance indicator definition and

then explain the PUF metric that we propose. All of them are based on *pdfs* of the measured delays.

4.3.1 Notations

The notations used to define our metrics are listed in Table 4.1.

TABLE 4.1 – Notations used to define the metrics.

Notation	Explanation
M	Number of delay elements on a delay chain in the PUF.
N	Number of serially implemented delay chains in the PUF.
L	Number of studied PUFs.
T	Number of carried tests to evaluate the steadiness of a PUF.
c_i^j	Control bit of the j^{th} element on the i^{th} delay chain of the PUF. If $c_i^j = 0$, the multiplexer path is the top else the bottom.
d^j_{i,c_i^j}	Delay of the delay element j of the delay chain i when applying the control bit c_i^j.

In the case of the arbiter PUF structure which is composed of only two identical and parallel delay chains which are identically controlled $N = 1$, the PUF response is directly related to the delay difference of the bottom and the upper delay chains. We consider that both the bottom ($1B$) and the upper ($1U$) delay chains of the arbiter PUF are seen as a single delay chain. We can denote them $c_1^j = c^j$ and $d_0^j = d^j_{1U,0} - d^j_{1B,0}$ to express the delay difference between the upper and the bottom delay chains of the delay element j when $c^j = 0$, and $d_1^j = d^j_{1U,1} - d^j_{1B,1}$ to show the delay difference between the upper and the bottom delay chains of the delay element j when $c^j = 1$.

However, as seen in Section 2.2.2, the loop PUF structure is composed of $N > 2$ identical delay chains. They are serially placed and are individually controlled. The loop PUF response depends on the delay difference of delay chains when they are differently controlled but use the same set of control words (same challenge) as explained by Equation(2.1). Using the same challenge set, the loop PUF can generate N bit responses. We note then that $c_{1,i} \neq c_{2,i} \neq \cdots \neq c_{N,i}$.

4.3.2 Metrics Computation

In this section we define and explain the PUF metrics which are based on *pdf* of the measured delays.

4.3.2.1 Randomness

We define the randomness on the PUF as its ability to produce as many zeros as ones. Then, the randomness can be expressed as

$$\mathsf{Rand} = 1 - |\mathsf{Pr}(\mathsf{ID} = 0) - \mathsf{Pr}(\mathsf{ID} = 1)|. \tag{4.2}$$

As defined in Equation (4.2), a randomness of 100% corresponds to a PUF with good randomness properties. It produces zeros and ones with the same probability. The randomness value depends on the error probability when the Gaussian distribution D_R of delay differences of involved delay elements to generate the PUF response is not perfectly centred in 0.

Hereafter, we detail the steps to obtain our metric for an arbiter PUF structure. For the arbiter PUFs, the probabilities to obtain an ID at 0 and 1 are expressed in Equation (4.3).

$$\mathsf{Pr(ID = 0) = 1 - Pr(ID = 1) = Pr(\sum_{j=1}^{M} d^j_{c^j} \ < \ 0).} \tag{4.3}$$

To scan the 2^M challenge possibilities, we consider the two complementary challenges for each delay element. Then, we notice

$$\sum_{j=1}^{M} d^j_{c_j} + \sum_{j=1}^{M} d^j_{\overline{c^j}} = \sum_{j=1}^{M} (d^j_0 + d^j_1).$$

To compute the probability to have an ID equal to 0, we propose to study the distribution D_R which represents the probability density function of $\sum_{j=1}^{M}(d_0^j + d_1^j)$. Figure 4.2 represents the *pdf* of the D_R distribution which is built with the measurements of $\sum_{j=1}^{M}(d_0^j + d_1^j)$ and the variance $M \cdot \sigma^2$. We consider $E(D_R)$ its mean value. It can be expressed as follows.

$$\mathsf{E(D_R) = \frac{1}{2} \sum_{j=1}^{M} (d^j_0 + d^j_1).}$$

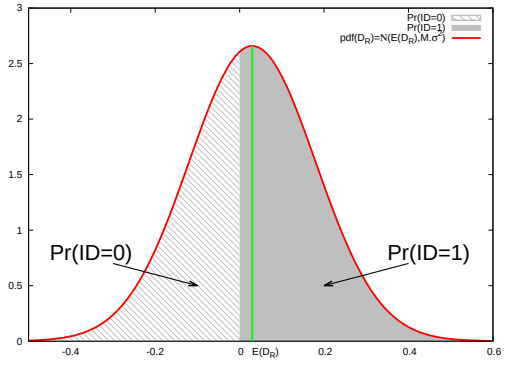


FIGURE 4.2 – The pdf of D_R distribution.

The probability to have an ID equal to 0 corresponds to have a value x under the threshold 0. Therefore, we suggest to compute this probability using the error

function as defined in Section 4.2. Form Equation (4.1), we obtain

$$\mathsf{Pr(ID = 0) = Pr(D_R < 0) = \frac{1}{2}(1 \ - \ erf(\frac{E(D_R)}{\sigma\sqrt{2 \cdot M}}))}.$$

Combining Equation (4.1) and Equation (4.2), the randomness expression for an arbiter PUF is given by

$$\mathsf{Rand_{arbiter} = 1 - |erf(\frac{E(D_R)}{\sigma\sqrt{2 \cdot M}})|}.$$

In summary, to evaluate the randomness quality of a delay based PUF such as arbiter PUFs or loop PUFs without the need of a lot of tries and also at design stage, four steps are needed :

1. Measure delays or frequencies of basic elements.
2. Draw the distribution of the probability density function of delays/frequencies measurements.
3. Compute the probability to have an ID equal to zero/one using error function (Equation (4.1)).
4. Calculate the randomness metric such as represented in Equation (4.2).

4.3.2.2 Uniqueness

The uniqueness performance indicator describes the ability of the PUF to produce different responses if implemented in different place of an IC (intra-Uniqueness) or in the same placement on different ICs (inter-Uniqueness). To evaluate the intra-uniqueness performance of a PUF, we need to duplicate the PUF structure several times on the same IC. However, the evaluation of the inter-uniqueness PUF performance needs the implementation of the same function structure on different devices (a large number). We present then only the intra-uniqueness evaluation. Note that the same method is adopted to evaluate the inter-uniqueness PUF performance.

We consider L identical PUF structures implemented in different places in the same device.

To evaluate the PUF performance at the design stage by measuring delays of basic elements, we propose to compare the distribution $D_{i,j}^{L}$ of the delay elements placed in the same position in different PUFs for $i \in \{1, N\}$ and $j \in \{1, M\}$ to the global distribution D of all delay elements of all studied PUFs. We consider that for the arbiter PUF the top and bottom basic delay elements, when placed in the same range on both delay chains, constitute one delay element.

The distribution $D_{i,j}^{L}$ represents :

– for the arbiter PUF, the distribution of the delay difference $(d_0^j - d_1^j)$ of L elements in the same range $j \in [1, M]$.

– for the loop PUF, the delay distribution of the delay difference $d^j_{i,0} - d^j_{i,1}$ of the element j of the delay chain i for the L different PUFs, with $j \in [1, M]$ and $i \in [1, N]$.

If some delay elements are biased, the comparison between their respective distribution will mitigate their resemblance. Hence they may influence the PUF performance.

The uniqueness value of delay based PUF is the mean of the $M \cdot N$ probabilities corresponding to $M \cdot N$ comparisons of distributions : We propose to express it as

$$\mathsf{Uniq} = \frac{1}{\mathsf{M} \cdot \mathsf{N}} \sum_{\mathsf{i}=1}^{\mathsf{N}} \sum_{\mathsf{j}=1}^{\mathsf{M}} \mathsf{Pr}(\mathsf{D}^{\mathsf{L}}_{\mathsf{i},\mathsf{j}} = \mathsf{D}). \tag{4.4}$$

A PUF have a good uniqueness property when all compared distributions are confused. Then it presents a uniqueness closed to 100%. We propose to compare each two distributions by computing their common area as illustrated in Figure 4.3.

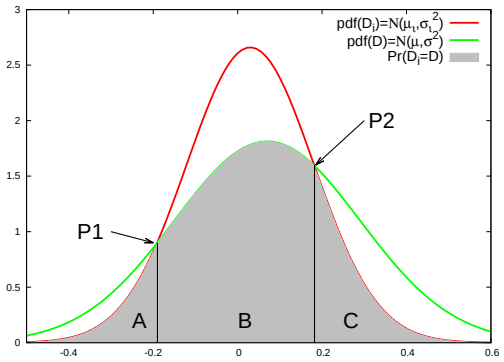


FIGURE 4.3 – Example of the comparison between two Gaussian distributions

To compute the common area of two distributions, we have to know the x-coordinate of the intersection points $P_{1,i,j}$ and $P_{2,i,j}$. After some algebra, the intersection points x-coordinate can be written as

$$P_{1,i,j} = \frac{-(\mu\sigma^2_{i,j}+\mu_{i,j}\sigma^2)-\sqrt{(\mu\sigma^2_{i,j}+\mu_{i,j}\sigma^2)^2-(\sigma^2_{i,j}-\sigma^2)[\mu^2\sigma^2_{i,j}-\mu^2_{i,j}\sigma^2-2\sigma^2\sigma^2_{i,j}Ln(\sigma_{i,j}/\sigma)]}}{\sigma^2_{i,j}-\sigma^2}$$

$$P_{2,i,j} = \frac{-(\mu\sigma^2_{i,j}+\mu_{i,j}\sigma^2)+\sqrt{(\mu\sigma^2_{i,j}+\mu_{i,j}\sigma^2)^2-(\sigma^2_{i,j}-\sigma^2)[\mu^2\sigma^2_{i,j}-\mu^2_{i,j}\sigma^2-2\sigma^2\sigma^2_{i,j}Ln(\sigma_{i,j}/\sigma)]}}{\sigma^2_{i,j}-\sigma^2}.$$

After computing the x-coordinate of the intersection points, we have to calculate the common area. This common area can be divided into three area : A, B and C as represented in Figure 4.3. If $P_{1,i,j} \neq P_{2,i,j}$ then it derives two cases, $\sigma < \sigma_{i,j}$ and $\sigma > \sigma_{i,j}$.

If we consider that $x_1 \neq x_2$ with

$x_1 = min(P_{1,i,j}, P_{2,i,j})$, and $x_2 = max(P_{1,i,j}, P_{2,i,j})$.

Suppose that F_i is the pdf of the $D^L_{i,j}$ distribution, and F is the pdf of the global distribution D. Using Equation (4.1), we can define

$F_{i,j}(t) = Pr(x < t) = \frac{1}{2}(1 + erf(\frac{t-\mu_{i,j}}{\sigma_{i,j}\sqrt{2}}))$,

and,

$F(t) = Pr(x < t) = \frac{1}{2}(1 + erf(\frac{t-\mu}{\sigma\sqrt{2}}))$.

Thus, if $\sigma > \sigma_{i,j}$, the common area of the two compared distributions is

$$\mathsf{A} + \mathsf{B} + \mathsf{C} = 1 + \mathsf{F_{i,j}}(\mathsf{x_1}) - \mathsf{F_{i,j}}(\mathsf{x_2}) + \mathsf{F}(\mathsf{x_2}) - \mathsf{F}(\mathsf{x_1}).$$

If $\sigma < \sigma_{i,j}$, the common area of the two compared distribution is expressed by :

$$\mathsf{A} + \mathsf{B} + \mathsf{C} = 1 + \mathsf{F_{i,j}}(\mathsf{x_2}) - \mathsf{F_{i,j}}(\mathsf{x_1}) + \mathsf{F}(\mathsf{x_1}) - \mathsf{F}(\mathsf{x_2}).$$

Figure 4.3 gives an example when $\sigma_{i,j} > \sigma$ and $P_{1,i,j} < P_{2,i,j}$. Then, in this case, the common area of the two distributions is expressed as follows :

$$\begin{aligned} \mathsf{Pr}(\mathsf{D^L_{i,j}} = \mathsf{D}) = 1 + \frac{1}{2}(\mathrm{erf}(\frac{P_{1,i,j} - \mu}{\sqrt{2}\sigma}) - \mathrm{erf}(\frac{P_{2,i,j} - \mu}{\sqrt{2}\sigma})) \\ + \frac{1}{2}(\mathrm{erf}(\frac{P_{2,i,j} - \mu_{i,j}}{\sqrt{2}\sigma_{i,j}}) - \mathrm{erf}(\frac{P_{1,i,j} - \mu_{i,j}}{\sqrt{2}\sigma_{i,j}})). \end{aligned} \tag{4.5}$$

In summary, to evaluate the uniqueness property of a PUF by measuring physical values, we need to :

1. Measure the L delay differences of delay elements in the same range j of the delay chain j, with $j \in \{1, M\}$ and $i \in \{1, N\}$ (whether implemented in the same device or not).
2. Draw the global distribution D of the $M \cdot N \cdot L$ delay elements.
3. Draw normal distributions $D^L_{i,j}$ for $j \in \{1, M\}$ and $i \in \{1, N\}$.
4. Compare each distribution $D^L_{i,j}$ to the global distribution D by computing the common area.

4.3.2.3 Steadiness

The third performance indicator is the steadiness. It allows users to evaluate the ability of the PUF to produce always the same response even in different environmental conditions. We propose a metric based on the evaluation of physical values of delay elements of the PUF to evaluate it. The purpose is to evaluate the delay difference of each delay element. The closer is the delays difference to 0, the greater is the probability that the PUF response is wrong.

Therefore, every delay difference of a delay element j of the delay chain i, $(d^j_{i,0} - d^j_{i,1})$ with $j \in \{1, M\}$ and $i \in \{1, N\}$, is measured T times. Then, we consider :

- $M \cdot N$ normal distributions $D_{i,j}$ of the measured delay differences. Their mean values are centred respectively in $E(d^j_{i,0} - d^j_{i,1})$ with a variance S^2 common for all delay elements (based on experimental results).
- The global distribution D. It corresponds to the distribution of the mean values $E(d^j_{i,0} - d^j_{i,1})$, centred in 0 (in case of ideal randomness) with a variance σ^2. However, for the loop PUF, if we take into account the constraint of H_{max} given is Section 2.1.2, its delay variance will be $\sigma'^2 = H_{max} \cdot \sigma^2$.

To evaluate the steadiness, we propose in fact, to calculate the area present in $[-\lambda, \lambda]$ for all the $D_{i,j}$ distributions as shown in Figure 4.4. The value of λ is indeed chosen such that the probability to have an error is close to zero. For instance if we choose $\lambda = 3 \cdot S$, this corresponds to a quintile which gives 0.23% of error which is very low.

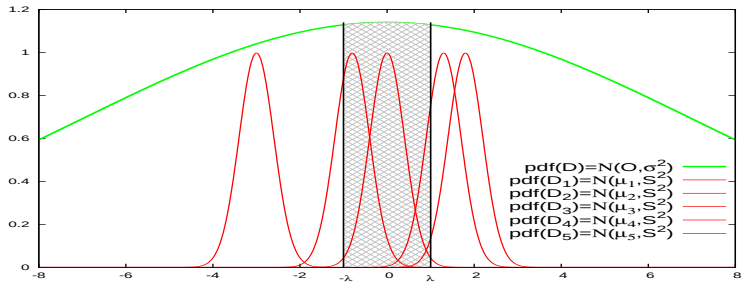


FIGURE 4.4 – Distributions $D_{i,j}$ after T measurements

Then, the steadiness metric is the product of probabilities where the first term is the probability to have an error when the element is in the λ window multiplied by the probability that this element is in the λ window. This can be expressed by the following equation :

$$Steadiness = Pr(error|delay < |\lambda|) \cdot Pr(element < |\lambda|). \tag{4.6}$$

A PUF presents a good steadiness property when its steadiness value is close to 100%. The probability of $(error|delay < |\lambda|)$ is an integral on the *erf* function which can be approximated with the Taylor's series at the second order.

$$\mathsf{Pr}(\mathsf{error}|\mathsf{delay} < |\lambda|) = \frac{1}{2\lambda} \int_0^{\lambda} (1 - \mathrm{erf}(\frac{x}{S\sqrt{2}}))dx \tag{4.7}$$

$$= \frac{1}{6S} \int_0^{3S} (1 - \frac{2}{\sqrt{\pi}}(\frac{x}{S\sqrt{2}} - \frac{x^3}{3(S\sqrt{2})^3}))dx$$

$$= \frac{4\sqrt{2\pi} - 3}{8\sqrt{2\pi}}. \tag{4.8}$$

Using Equation (4.1), the probability that the element is between $-\lambda$ and λ for an arbiter and a loop PUF is, respectively

$$\mathsf{Pr}(\mathsf{element} < |\lambda|) \;=\; \mathrm{erf}(\frac{\lambda}{\sigma\sqrt{2}}), \tag{4.9}$$

and

$$\mathsf{Pr}(\mathsf{element} < |\lambda|) \;=\; \mathrm{erf}(\frac{\lambda}{\sigma\sqrt{2 \cdot \mathsf{H}_{\mathsf{max}}}}). \tag{4.10}$$

Finally, using Equations (4.6), (4.7) and (4.10), the steadiness of an arbiter and a loop PUF has these probability expressions which depends merely on the ratio S/σ :

$$\mathsf{Stead}_{\mathsf{arbiter}} \;=\; \frac{4\sqrt{2\pi} - 3}{8\sqrt{2\pi}} \times \frac{\mathsf{S}}{\sigma}\sqrt{\frac{2}{\pi}};$$

$$\mathsf{Stead}_{\mathsf{loop}} \;=\; \frac{4\sqrt{2\pi} - 3}{8\sqrt{2\pi}} \times \frac{\mathsf{S}}{\sigma\sqrt{\mathsf{H}_{\mathsf{max}}}}\sqrt{\frac{2}{\pi}}.$$

In summary, to characterize the steadiness of delay based PUFs by evaluating physical values of delay elements, we need four steps :

1. Test T times each delay difference between the M delay elements on each delay chain $i \in [1, N]$ when applying the control bit 0 and 1.
2. Trace the $M \cdot N$ distributions $D_{i,j}$ of the delay differences.
3. Trace the global distribution D corresponding of the distribution of mean values of delays differences.
4. Compute the steadiness value using Equation (4.6).

4.4 Experiments and Results

To test our characterization method, we first implement an arbiter PUF as illustrated by Figure 2.9 on an ALTERA FPGA platform. Then we evaluate its performance using our proposed method. Second we implement the loop PUF into the same platform and then we evaluate its performance using the same

proposed method. To be evaluated using our proposed method, the arbiter PUF design should be adapted. We have to oscillate the arbiter PUF structure in order to measure the frequency of each delay element. However, the structure of the loop PUF does not need any changes. The measurements of the frequencies of all delay elements can be obtained using the same design. Finally, in order to compare the proposed method to the Hori method, we propose to compare the performance of both structures (loop and arbiter) when evaluated using both metrics. For that, the experiments are performed using the ASIC platforms (See Chapter 3).

In this section, we first present how do we implement the arbiter structure for the evaluation and then we show the experimental results for both arbiter and loop PUFs when designed on the ALTERA FPGA.

4.4.1 Arbiter PUF Design on ALTERA FPGAs

4.4.1.1 Experiments

The evaluation of the arbiter PUF structure have been carried out in an ALTERA FPGA CYCLONE II EP2C35F672. To evaluate the intra-device performance of the PUF, we reproduce its structure $L = 16$ times. We consider an arbiter PUF with $M = 8$ delay elements per each delay chain. Since the arbiter PUF response is correlated with the delay difference between two parallel delay chains, the placement and routing of the 32 delay chains has to be well done. All delay chains have to be exactly routed and placed. This is possible on the ALTERA FPGAs by including placement constraints. Therefore, we choose to implement all delay chains using the same column and different rows as shown in Figure 4.5. To be sure that all delay chains are routed identically, we used the Time Quest as a timing analyser.

For the evaluation of the arbiter PUF, we have to compute the delay of each delay element. Therefore, the delay chains are enclosed to form a ring oscillator as shown in Figure 4.6.

We use the method described in Section 2.2.1.1 to measure the period or the frequency of each delay element. To evaluate the uniqueness and the steadiness of the arbiter PUF, we have to measure the delay differences $(d^j_{i,0} - d^j_{i,1})$ of every delay element $j \in [1, M]$ of the delay chain i. As denoted before, the two parallel delay chains of the arbiter PUF are seen as a single delay chain $(i = 1)$. Then, to measure the delay of the j^{th} delay element, the challenge bit $C_{1,j}$ and the control signal *choix_osc* are driven alternately, when the others challenge bits remain constant. Figure 4.7 shows the delays which are involved in the measurements associated to the four possible combinations of $C_{1,j}$ and *choix_osc*. As the measurement is differential $(d^j_{1,0} - d^j_{1,1} = d^j_{a0}$ - d^j_{a1} - d^j_{b0} + d^j_{b1}), the external delays, as α and β, are eliminated. In our studies, we consider $T = 128$ experiments to measure the noise variance S^2 needed for the steadiness.

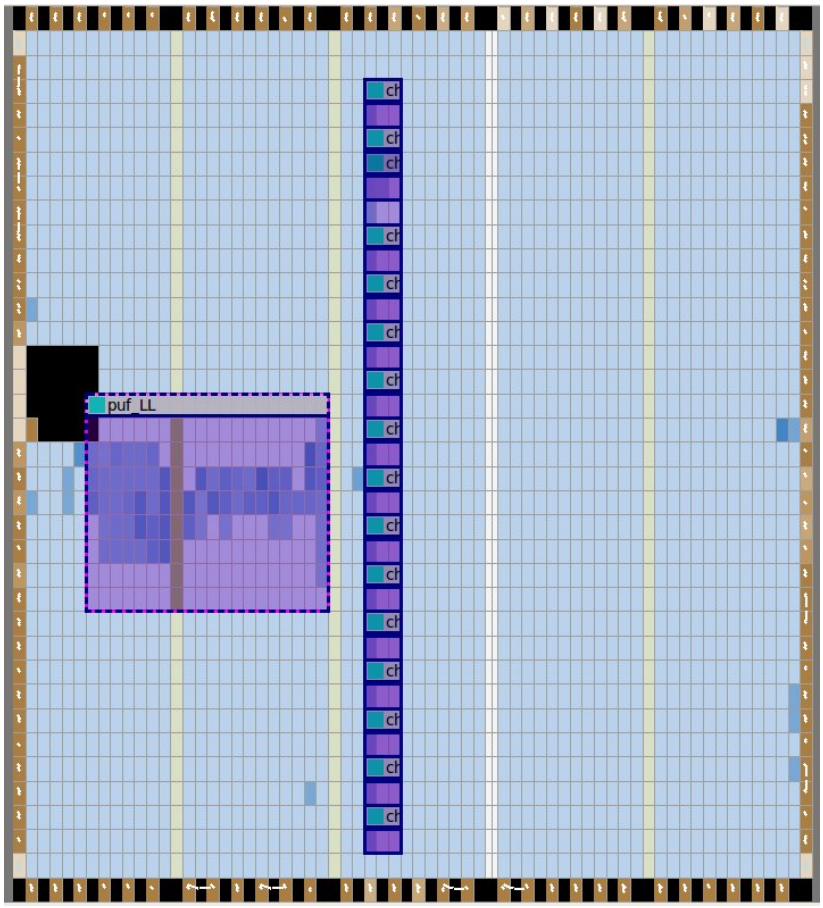


FIGURE 4.5 – Arbiter PUFs layout.

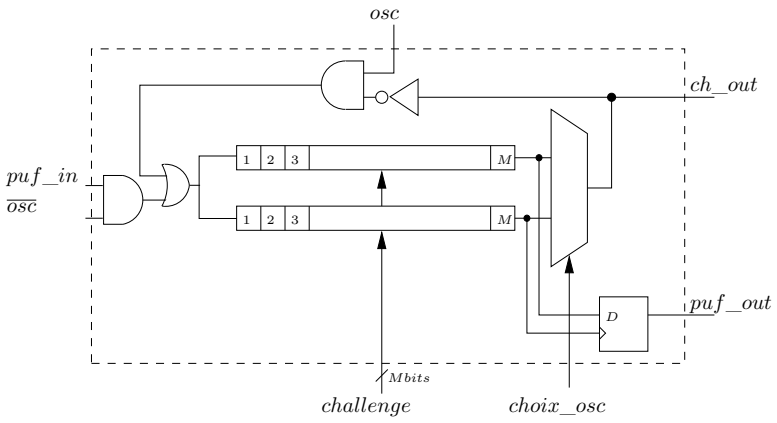


FIGURE 4.6 – Implementation design.

For the randomness, the measurement of $E(D_R)$ is global, the challenge bits are set alternately all at '1' and all at '0'. We consider that the variance σ^2

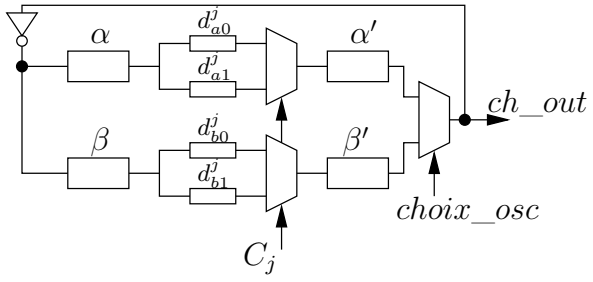


FIGURE 4.7 – Measurement of element j.

of$(d^j_{1,0} - d^j_{1,1})$ is equal to the variance of $(d^j_{1,0} + d^j_{1,1})$ needed for the randomness evaluation.

4.4.1.2 Results

In this section, we present results of the evaluation of the arbiter PUF performance using our characterization method as defined in Section 4.3.2. Note that the PUF has the optimum quality when the metric are closed to 100%. To evaluate the steadiness of the arbiter PUF, experiences has been carried out in a stable environment, with a temperature $Temp = 23°C$. The measurement variance are $\sigma = 17.2251\ ps$ and $S = 0.61ps$. Table 4.2 gives the evaluation results of an arbiter PUF based on two delay chains.

TABLE 4.2 – The experimental results of the intra-device evaluation of the arbiter PUF.

Performance indicator	**Result**
Randomness	0%.
Intra-Uniqueness	97.73%.
Steadiness	99.07%.

Using our metrics, the implemented arbiter PUF has a good intra-uniqueness property. However, we see that the implemented PUF structure is biased. The PUF response is not random at all. This shows that the constraints of placement are not sufficient to equilibrate the two independent delay lines. This influences positively the steadiness of the PUF which presents a stable response when tested T times at nominal operating conditions.

4.4.2 Loop PUF Design on ALTERA FPGAs

In order to measure the intra-uniqueness and steadiness using the proposed metrics, experiments have been carried out on a design with 8 loop PUFs embed-

ded in a CYCLONE II FPGA. Each loop PUF has $N = 3$ chains with $M = 8$ delay elements. Note that all loop PUFs have to be identically designed and also, all delay chains of the same loop PUF have to be identical. Figure 4.8 shows the layout of the loop PUFs where all delay chain are placed in the same row in order to be balanced.

The loop PUF randomness is theoretically maximal as the IDs are built from delay difference. Thus if we consider the complementary challenges, the ID results are complementary. For instance, for $N = 2$ with the challenge words C_1, C_2 the ID is :

$$ID_{C_1,C_2} = sign(D_{C_1C_2} - D_{C_2C_1})$$

which is complementary to the ID with challenge words C_2, C_1 :

$$ID_{C_2,C_1} = sign(D_{C_2C_1} - D_{C_1C_2})$$

As these two IDs are correlated, it does not make sense to use both a challenge and its complement. If only one is used randomly, either the chosen challenge or its complement, the randomness should remain statistically perfect.

$$Randomness_{loop} \approx 100\ \%\,.$$

We consider that the challenge set sent to the PUF is $C_1, C_2, C_3 = 0, 0, 2^j$, with j being the index of the element. The loop PUF controller takes this challenge set to operate the rotations and gives the difference of delay for each $M \times N$ elements.

Table 4.3 gives the results for 8 loop PUFs with M=8 and N=3. The number of $T = 128$ tries are performed to study the steadiness.

TABLE 4.3 – The experimental results of the intra-device evaluation of the loop PUF.

Performance indicator	**Loop PUF**
Randomness	$\approx 100\%$
Intra-Uniqueness	95%
Steadiness	98.7%

The loop PUF is naturally random. Although measurements have been done with challenges whose $H = 1$ (refer to Equation 2.2) and not $H_{max} = 2$ for $N = 3$, we had a good uniqueness 95%. Also, in normal environmental conditions, the loop PUF presents a high steadiness performances (>98.7%).

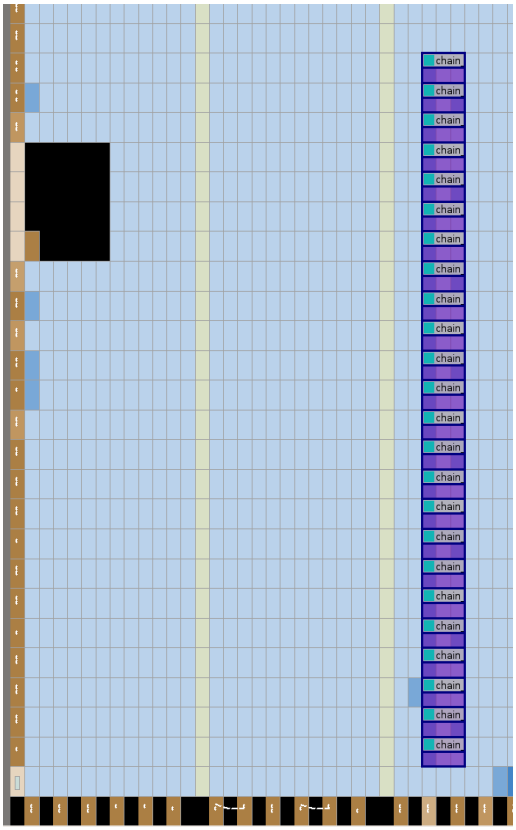


FIGURE 4.8 – Layout of 8 loop PUFs with $N = 3$ in CYCLONE II.

4.4.3 PUF_{mix} Design on ASIC

In this section, we present the evaluation results of the PUF_{mix} design on ASIC. Table 4.4 shows the performance evaluation results of both arbiter and loop PUFs when evaluated using the Hori (Section 2.2.4.3) and our method. We note that either the arbiter or the loop PUF presents similar uniqueness and steadiness results when evaluated using both methods. However, when evaluating the randomness of the arbiter PUF we note a quite big difference between both methods. The randomness of the arbiter PUF when evaluated using the Hori method is around 32%. And when the same structure is evaluated using our proposed method, it presents a randomness around 3%. This difference can be explained by the precision of our method, the results shown do not depend on a specific challenge set. However, the results of the Hori method are specific to a set of challenges. The complete evaluation of PUF performances using the Hori method needs the scan of all possible challenges which is practically impossible for both arbiter and loop PUFs. Therefore, we randomly choose a set of 1024 challenges for the evaluation of the arbiter PUF. For the loop PUF also we select a set of 1024 challenges that verify the defined constrains in Section2.1.2.

TABLE 4.4 – Arbiter and loop PUF performance results when evaluated using Hori metrics and our metrics.

	Arbiter PUF			**Loop PUF**		
Metrics	**Rand.**	**Uniq.**	**Stead.**	**Rand.**	**Uniq.**	**Stead.**
Hori	32.56%	88.53%	91.86%	98.97%	89.21%	98.26%
Cherif	3.17%	96.47%	84.93%	100%	97.07%	95.49%

However, using our metrics the number of needed tests depends linearly on the number of delay elements (serially designed) on the PUF design which is very small comparing the number of the tests needed using the Hori method. Moreover, our metrics can be used at design stage to evaluate the PUF performances. It can compare with model dispersion as for the "Pelgrom" model at design stage.

4.5 Conclusion

In this chapter, we first presented a novel method to evaluate silicon delay based PUF. The three proposed metrics are based on probabilities. To the contrary to existing methods, they evaluate physical values of delay elements. Therefore, our method can be used at design stage by electrical simulation of delays.

Next, Tests have been carried out in ALTERA FPGA and ASIC. Two PUF structures are evaluated : the arbiter and the loop PUFs. First, we evaluated 16 arbiter PUFs based on two delay chains when implemented in ALTERA Cyclone-II FPGA. We presented the method we used to measure delay element period and the way that we add extra logic components for measurements. Results underline the weakness of this arbiter PUF implementation to produce as many 1 as 0 which influence positively the PUF steadiness. However, we see that this implementation propose a good intra-Uniqueness property.

Second, we evaluated 8 loop PUFs based on three delay chains when implemented in ALTERA Cyclone-II FPGA. This PUF structure does not need any extra logic components for measurements. Results underline that the loop PUF presents high performance.

Finally, in order to compare our method to the characterization methods based on logical values of the PUF response (e.g. Hori metrics), we proposed to evaluate the loop and the arbiter PUF when implemented on ASIC platform using both metrics. Results show that both methods present similar results for the uniqueness and the steadiness. And for the randomness, the proposed method is much more precise since the results do not depend on the selected challenge set.

The advantages of our metrics that they do not need a huge number of tests to evaluate the PUF performances. They also can be used at design stage. However

they are only used to evaluate delay PUFs, and the evaluation have to be done by the designers.

In the next chapter, we present the way to use the proposed loop PUF for the generation of cryptographic keys and for device authentication purposes.

Chapitre 5

Loop PUF : Device Authentication and Cryptographic Key Generation

The topic addressed in this chapter is related to the applications of PUF. We first present the motivations to use the "loop PUF" for device authentication and for cryptographic key generation. Indeed, it has interesting properties which could contribute to increase the reliability of these critical applications. The device authentication procedure using the "loop PUF" is presented, along with experimental results when tested on ASIC platforms. Then we describe the key generation procedure which has been developed for the loop PUF. It relies on methods to enhance the reliability, including the increase of measurement time, the detection of unreliable bits and the use of low-cost error correction mechanisms. Then, we show and discuss the obtained results on different PUFs designed in 65nm ASICs technology.

5.1 Motivation

Due to the combination of their properties of uniqueness and steadiness, PUFs are presented as an innovative primitive to derive secret from complex physical characteristics of ICs. They are proposed to be used for low cost authentication of ICs and the generation of cryptographic keys. However, as seen in previous chapters, most of the proposed PUFs present non ideal performance results. Their responses require a post processing treatment to be used for either device authentication or key generation purposes.

Suh et al. [SD07] propose methods to use the ring-oscillator PUFs for low cost authentication and cryptographic key generation. For the authentication process, the authors suggest to use the CRP protocol as a methodology to authenticate a device. They propose first to record a challenge response table. And then at each authentication operation, the trusted party has to select only one recorded and not previously used challenge. Finally, to check the authenticity of the IC, the responses of the PUFs for the same selected challenge have to meet **exactly** the response saved on the recorded table. Using this method, we consider that the PUF response is perfect and no errors can occur, which is not true since the error probability of the proposed PUF is not null. The device authentication procedure is not a critical application and then a minimum margin of error has to be considered. However, the generation of cryptographic keys is a very critical application and need to be error-free. Hence it is not possible to use directly the PUF response as a key without an adapted post processing. A lightweight error correction is generally recommended in order to ensure both a very low error probability and a low complexity key generation system. But a trade-off has to be found as the higher the PUF reliability performance, the lower the complexity of the post processing block. In this chapter we are interested on the performance of the loop PUF for the authentication and key generation applications. The loop PUF presents indeed two important characteristics :

- **Precise output value**. The Loop PUF outputs an integer value which represents the delay or the frequency of the loop during a fixed period. From this output, we can either generate a binary response (as for the others PUFs on the literature) as suggested in Chapter 2 or use directly the integer output which is much more precise and significant. In this chapter, we propose a method that takes advantage from this integer output for device authentication purpose.
- **Huge number of challenges**. As seen in Chapter 2, the reliability of the loop PUF increases when we choose the best challenge set. This allows us to use the loop PUF in more stringent applications as the cryptographic key generation with a low cost. The loop PUF provides already reliable response and then the post processing block needed is not cost effective. Hereafter, we describe a methodology to use the loop PUF for low cost

generation of cryptographic keys.

5.2 Loop PUF for Authentication

In this section, we are interested on the ability of the loop PUF response to be used for device authentication purposes. The proposed method is based on the measurement of physical values of delay elements. These physical values are used to authenticate devices since they are much more precise than the binary response of the loop PUF. Unlike the Suh et al. [SD07] method, in our case, we do not use the CRP protocol for authentication. In our method, a device is identified by its single signature. The proposed method can be divided into two principle steps :

1. The Learning step.
2. The authentication step.

Hereafter, we detail the proposed loop PUF-based authentication method.

5.2.1 PUF-based IC Analysis

The main rationale of this step is to generate the reference vectors which are used as a helper data, at the authentication step, to analyse the IC authenticity. It consists in measuring the delays of basic loop PUF delay elements. At the end of this stage, M reference vectors are saved, with M the number of basic delay elements included on a loop PUF delay chain. Therefore, we propose to :

1. measure the global delay d_0 of the loop PUF when all delay elements are set to '0' ($C_i^j = 0$ with $i \in [1, N]$ and $j \in [1, M]$).
2. measure separately the delay $d_{i,j}$ of each delay element j on each delay chain i when $C_i^j = 1$.
3. construct the measurement vectors Ref^j with $j \in [1, M]$ such that : $\mathsf{Ref^j} = [(\mathsf{d_{1,j}} - \mathsf{d_0}); (\mathsf{d_{2,j}} - \mathsf{d_0}); (\mathsf{d_{3,j}} - \mathsf{d_0}); (\mathsf{d_{i,j}} - \mathsf{d_0}); \cdots; (\mathsf{d_{N,j}} - \mathsf{d_0})]$.

We note that this procedure is performed only one time by the designer in order to generate the reference vectors which are compared to the generated vectors during the authentication step. The latter is detailed hereafter.

5.2.2 The Authentication Procedure

In order to verify the authenticity of its IC, the user activates the authentication procedure. It consists in checking if the generated measurement vectors are correlated or not with the references vectors. To do so, we propose to analyse the Pearson correlation coefficient. The proposed method is based on four essential steps. The three first steps are identical to those performed during the learning stage.

1. We measure the global delay d_0 of the loop PUF when all delay elements are set to '0' ($C_i^j = 0$ with $i \in [1, N]$ and $j \in [1, M]$).
2. We measure separately the delay $d_{i,j}$ of each delay elements j on each delay chain i when $C_i^j = 1$.
3. we construct the measurement vectors X^j with $j \in [1, M]$. $\mathsf{X^j} = [(\mathsf{d_{1,j}} - \mathsf{d_0}); (\mathsf{d_{2,j}} - \mathsf{d_0}); (\mathsf{d_{3,j}} - \mathsf{d_0}); (\mathsf{d_{i,j}} - \mathsf{d_0}); \cdots; (\mathsf{d_{N,j}} - \mathsf{d_0})]$.
4. We compute the Pearson coefficient (See Equation 5.1) between each normalized pair of reference and measurement vector $corr^j_{(Ref^j, X^j)}$ with Ref^j the reference vector already recorded.

The Pearson coefficient used as a metric to authenticate a device is given by

$$\mathsf{corr_{(Ref,X)}} = \frac{1}{\mathsf{M}} \sum_{\mathsf{j=1}}^{\mathsf{M}} \mathsf{corr^j_{(Ref,X)}} = \frac{1}{\mathsf{M}} \sum_{\mathsf{j=1}}^{\mathsf{M}} \frac{\sum_{\mathsf{i=1}}^{\mathsf{N}} (\mathsf{ref_i^j} - \hat{\mathsf{ref}}^{\mathsf{j}})(\mathsf{x_i^j} - \hat{\mathsf{x}}^{\mathsf{j}})}{\sigma_{\mathsf{ref}} - \sigma_{\mathsf{x}}}. \tag{5.1}$$

Using this metric for device authentication, we take into account both offset and scaling phenomenon that can affect the PUF response (e.g. by temperature variations). The correlation coefficient is equal to '1' if one of the variables is an increasing function of the other variable, to '-1' in when the function is decreasing. Intermediate values provide information on the degree of linear dependency between two variables. The correlation between variables is strong when the correlation coefficient value is closer to the extreme values '-1' and '1. A correlation coefficient of 0 means that the variables are not correlated.

5.2.3 Experimental Results

The tests have been carried out on 16 ASICs, each one embedding 49 loop PUFs. Hereafter we present first the authentication results when identifying a PUF among others embedded in the same IC (authentication intra-ASICs) either in ambient temperature or when varying it. Then we study the performance of our authentication metric to identify a PUF among others situated in the same location in identical ICs (authentication inter-ASICs) at nominal environmental conditions.

In our case, the studied loop PUF is composed of four delay chains ($N = 4$). Each one contains 16 delay elements ($M = 16$). As the routing of the 16 elements is identical, we can consider a unique delay chain composed of 64 elements ($N = 64$ and $M = 1$). We propose to repeat the steps (1), (2) and (3) (defined on the head of Section 5.2) $T = 128$ times in order to perform statistics on the reliability of the PUF response. We consider that the first test of the measurement vectors X^j as a reference vector referred to Ref^j. And then we compute the mean correlation coefficient as described on the fourth and last step of our authentication step which is done on software. We note that all the PUF measurements are

done using a fixed measurement window width with $t = 10$ (Figure 2.6).

Intra-device Authentication

In order to verify the ability of the loop PUF to be used for authentication purposes using the proposed method, we study the intra-ASIC correlation on the 16 ASICs.

Figure 5.1 shows the mean value of the correlation coefficients between all PUFs situated in the same device. We note that when we compare a PUF response with its responses (different tests), we have a high correlation (closed to 1). However, when the PUF responses are compared to those of another PUF existing in the same die, the worst case correlation coefficient does not exceed in its absolute value the 0.5 which is very low.

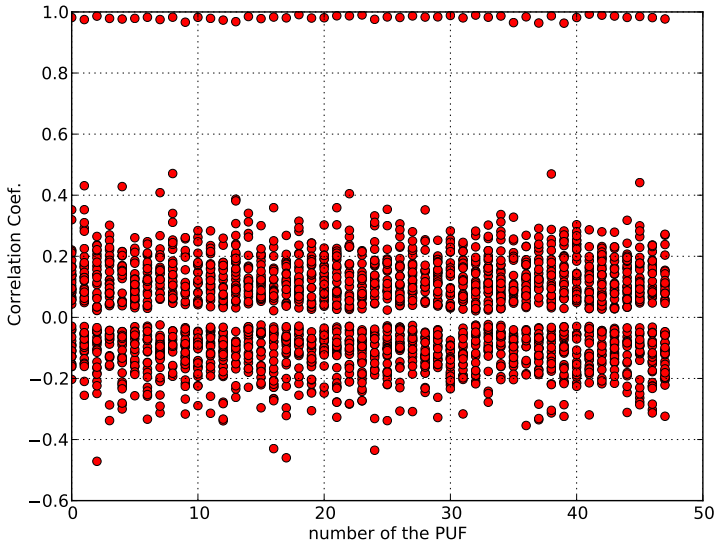
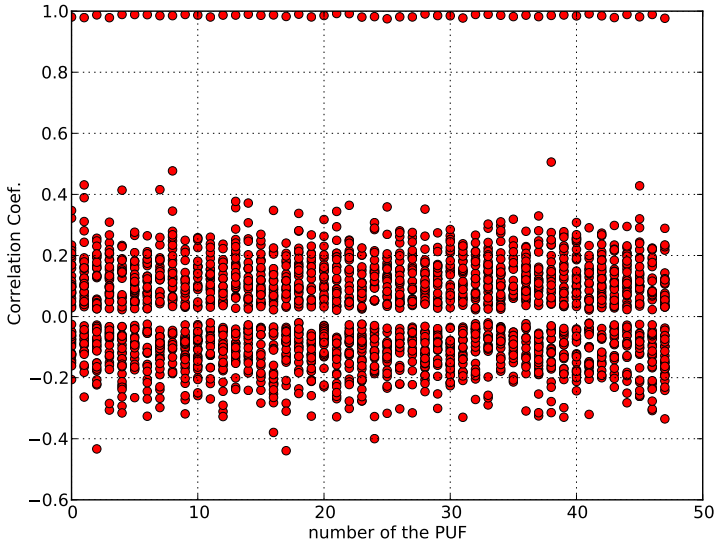


FIGURE 5.1 – Intra-ASIC mean correlation results.

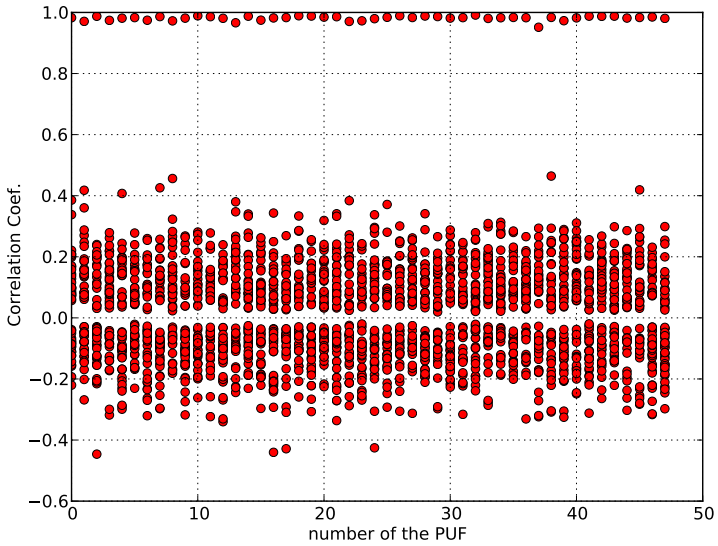
Even when decreasing the temperature to $0°C$ or increasing it to $60°C$, using our method (the Pearson correlation coefficient), the loop PUF presents good performance to be used for device authentication purposes. The correlation coefficient at different environmental conditions does not exceed in its absolute value the 0.5. This proves that our metric take into account the scaling phenomenon caused by the temperature variation. Figures 5.2a and 5.2b illustrate the intra-device authentication results when varying the temperature. We note that the results obtained at $60°C$ are the best. This can be explained by the fact that the reliability of the loop PUF increases with the temperature.

Inter-device Authentication

To evaluate the ability of the loop PUF to be used for device authentication using



(a) Temperature=0°C.



(b) Temperature=60°C.

FIGURE 5.2 – Intra-ASIC mean correlation results at different temperatures.

the correlation method, we propose to evaluate the correlation degree between each PUF (through its reference vector) and its equivalent (PUFs having the same place) in the other ASICs. Figure 5.3 shows that we are able to distinguish a PUF from another one even when placed in the same place on different ASICs. In the worst case, the correlation coefficient does not exceed in its absolute value

the 0.5 which let the error interval very large.

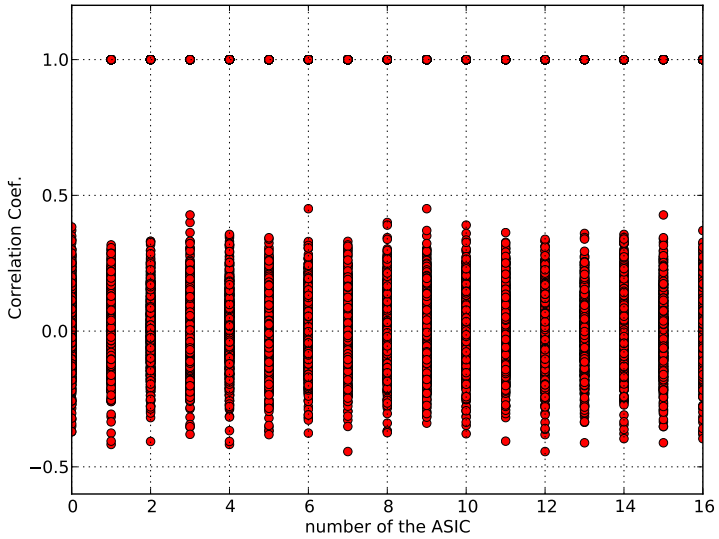


FIGURE 5.3 – Inter-ASICs correlation results.

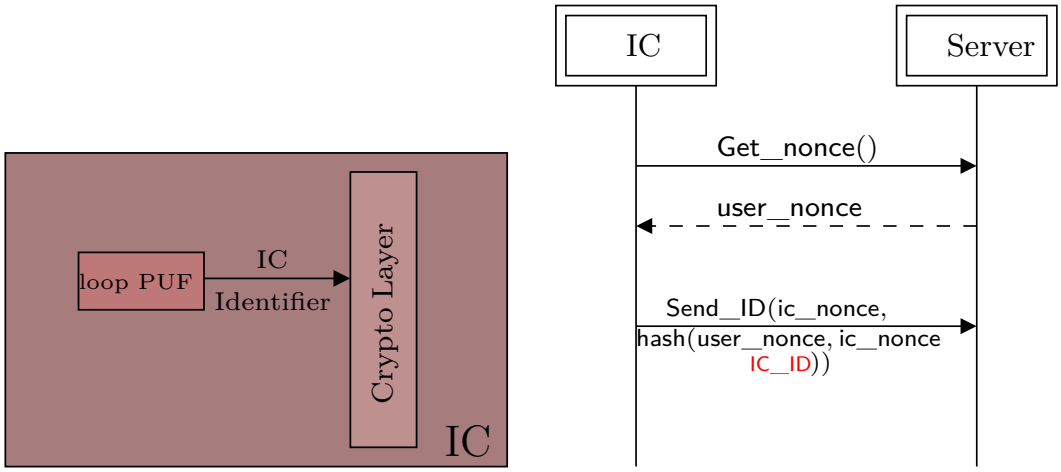
Due to the accuracy of the loop PUF output and the method that we propose, we are able to distinguish a PUF form other similar PUFs either implemented in the same ASIC or placed at the same place in different ASICs without post processing schemes and with a large error margin.

The authentication procedure is not time consuming. The authentication time is linearly dependent with the number of basic delay element of the loop PUF. In our case, since the loop PUF is composed of 64 basic delay elements, we need 65 measurements to authenticate a PUF. One measurement when all the control bits are set to '0', and 64 others when activating a single delay element per measurement. With a clock system frequency of 100Mhz, we are able to perform all needed measurements within $1.08ms = 65 * T_{puf}$ which is very low. The metric computation (the fourth step of our method) needs $0ms$ to be done. Then the overall authentication procedure of the PUF takes about $1.08ms$ for a loop PUF of 64 delay elements.

Discussion about the Robustness Against Attacks

In order to secure the transmission of the PUF response (IC identifier) from man in the middle attacks, replay attacks and modeling attacks, a cryptographic layer should be added to the PUF system. The traditional solution to thwart these attacks is to provide a secure challenge-response authentication protocol while exchanging the IC identifier. However this extra logic should not be too complex to harm the low-cost interest of the PUF. Figure 5.4a and Figure 5.4b

show an example of countermeasure against replay attacks. The cryptographic layer takes advantage of a **Hash** function and a **cryptographic nonce** authentication protocol. This protocol should be both ways if the server is not trusted.



(a) PUF authentication system. (b) Authentication protocol.

FIGURE 5.4 – Principle of the secured PUF authentication system.

In the next section we propose a cryptographic key generation method using the loop PUF structure.

5.3 Loop PUF for Cryptographic Key Generation

The main goal of this section is to present a novel procedure for key generation purposes using the loop PUF structure. The proposed method is divided into two steps : the profiling and the user key generation, and relies on five techniques to ensure a reliable low cost cryptographic key generation. In what follows, we first present the principles of the proposed method, the procedure is detailed, and then we discuss some experimental results.

5.3.1 Principles

Figure 5.5 illustrates the basic principle of the proposed method to use the Loop PUF response for the generation of a key bit.

For a given challenge and a specific measurement window, the PUF response is the number of oscillations occurring during a fixed measurement window.
In order to generate a cryptographic key bit, we propose to quantify the differences between two PUF responses T_1 and T_2, for two equivalent challenges ch1 and ch2, respectively. Two challenges are equivalent if their Hamming weight is the same, as all the delay elements have the same layout and interconnection

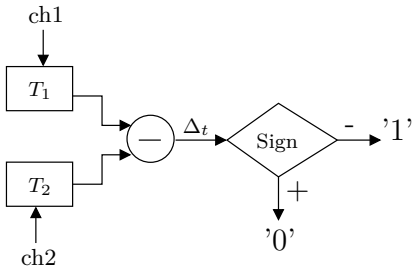


FIGURE 5.5 – Key bit generation.

routing. If $\Delta_T = T_1 - T_2$ is positive, then the key bit is quantified as a '0', '1' otherwise.

For a given challenge pair, the Δ_T measurement is highly dependant on the noise level, thus generating potential errors, especially if the Δ_T is close to 0. Figure 5.6 shows the *pdfs* of Δ_T for different challenge pairs. Therefore it is necessary to investigate the existing techniques and methods to improve the key bit reliability.

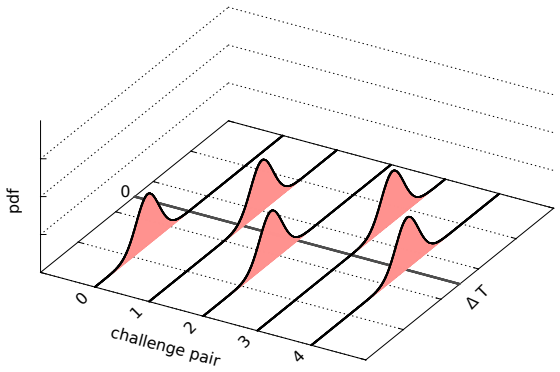


FIGURE 5.6 – Illustration of ΔT distributions.

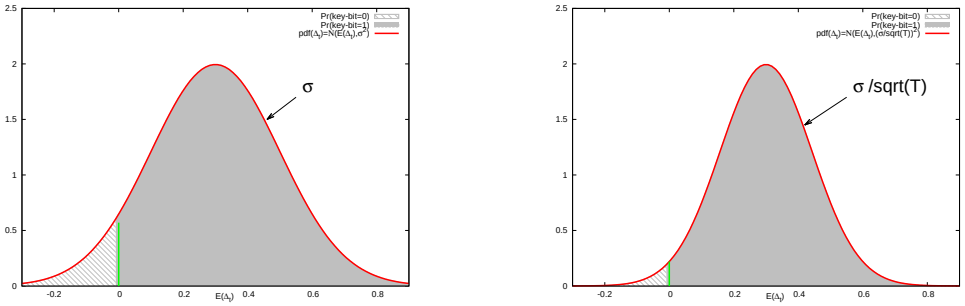
5.3.2 Reliability Improvement Techniques

In this section we present the available techniques to improve the error rate of the generated key using the loop PUF structure. Therefore, in order to enhance the reliability of the loop PUF response and then the key bit reliability, we distinguish five possible techniques. The first two techniques are directly related to the loop PUF structure.

1. **Selecting the challenge pairs**. The selected challenge pairs have to be as much different as possible. When the Hamming Distance $HD(ch_1, ch_2)$

increases, the delay difference ΔT statistically increases, thus minimizing the quantification error probability.

2. **Enlarging the PUF measurement window**. The precision of the PUF response is enhanced when we increase the measurement window width (Ww) parameter (which is equivalent to the input t value on Figure 2.6). It consists in increasing the delay difference ΔT, thus the Signal to Noise ratio SNR, and then enhancing the key bit reliability. However this also increases significantly the total time to generate the key. This can be problematic for applications requiring a fast response. Hence a trade-off has to be found.
3. **Increasing the number of tests** to generate a key bit. Figures 5.7a and 5.7b illustrate the evolution of the error probability when we increase the number of tests for the generation of each key bit.



(a) Standard deviation = σ.

(b) Standard deviation = $\sigma/\sqrt{T}$ after T iterations.

FIGURE 5.7 – Error rate evolution when increasing the number of tests.

The measurement error is minimized when the number of tests (itnb) increases. If the measured value of ΔT is inside the confidence interval of 3σ, σ being the standard deviation of the noise, there is 99.9% probability to have a reliable value. With T tests, the probability increases as the standard deviation of the noise decreases in $1/\sqrt{T}$ as shown in figures 5.7a and 5.7b.

4. **Removing the most unreliable bits**. When providing a statistical analysis of the key samples, the most unstable bits can be detected. Then they can be registered not to consider them for key generation. However, using this technique reduces the key length. In our case, we introduce a parameter referred to as "mnib" that define the maximum number of ignored bits. The BER increases significantly with the mnib parameter.
5. **Introducing a key correction procedure**. In order to generate a reliable cryptographic key from biometric data, Dodis et al. [DRS04] and [DORS08] present methods that allow a secure and reliable extraction of a key from the noisy biometric database. It is based on the concept of secure sketch

and fuzzy extraction, relying on Error Correcting Codes (ECC) and Hash function. They propose to correct the received data based on the information collected at the profiling step. Later on, Maes et al. [MVHV12] propose a PUF-based key generation procedure which is based on the same concept but implemented in FPGA with BCH ECC codes. In our case we propose to use a combination of the Hamming code and the Chase algorithm [Cha72] to correct the key bits. Therefore, we define two parameters : the maximum number of errors that our corrector is able to correct (mnce) and the number of unreliable bits considered (nub) that characterize our proposed key correction procedure. These two parameters influence the Binary Error rate (BER).

Table 5.1 shows the impact of the used parameters on the BER of the key bits.

TABLE 5.1 – The influence of some defined parameters on the BER.

Parameter	**Reliability improvement technique**	**Error Rate**
Ww ↗	enlarging the PUF measurement window.	↓↓
HD(ch1, ch2) ↗	Selecting the challenge pairs.	↓↓
itNb ↗	Increasing the number of tests to generate a key bit.	↓↓
mnib ↗	Removing the most unreliable bits.	↓↓
nub, mnce ↗	Introducing a key correction procedure.	↓↓

5.3.3 Profiling

The goal of this preliminary step is to compute the reference key and then to generate the helper data. It can be defined as an initialization or a learning step. The generated helper data is introduced at the user key generation step. Figure 5.8 shows the profiling stage flow of the proposed method.

As illustrated by Figure 5.8, the profiling stage can be divided into five essential steps which corresponds to the five technique enumerated in subsection 5.3.2 :

1. **Challenge selection** : according to the desired key length, we select the challenge pairs allowing us to have the best steadiness rate of the loop PUF.
2. **Measurement window size and noise estimation** : initial tests are needed to get the standard deviation σ of the measurement error. This information will be used at the key reliability analysis step.
3. **Increasing the number of tests if necessary** : this phase is also called **dynamic reliability enhancement**. For each key bit associated to a challenge pair, the tests are repeated until a certain reliability level dependant on σ.

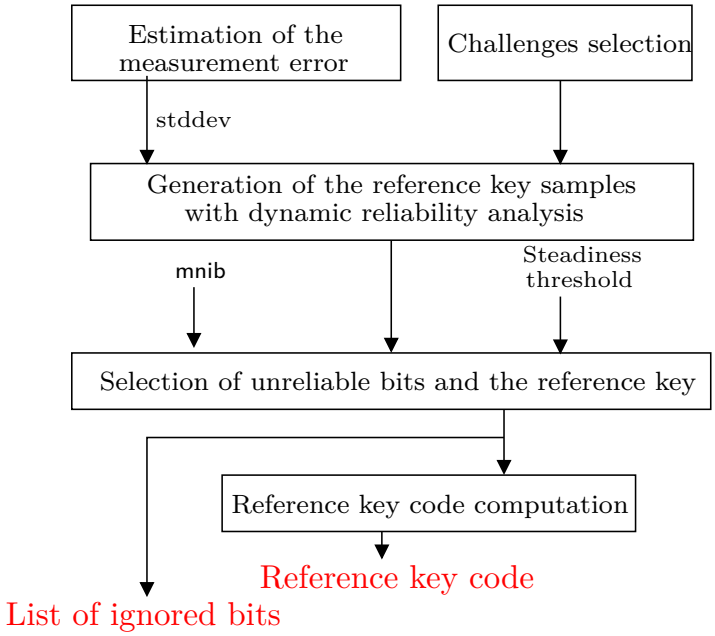


FIGURE 5.8 – Preliminary profiling flow.

4. **Unreliable bits selection** : This step generates the *reference key* after elimination of the unreliable bits. For each bit, we compute its steadiness rate which allows us, first, to select the most unstable bits and then to compute each bit value on the reference key. The saved information constitutes a part of the helper data which is used at the key generation process.
5. **Code generation for further correction** : the Computation of the *public* reference key code : based on the computed reference key and the list of ignored bits, we compute the public reference key code. The latter is used at the key correction step.

5.3.3.1 Challenge Selection

In order to generate a key of K bits, we have to select K challenge pairs. A key bit is deduced from the sign of the difference Δ_t between the PUF outputs when two delay chain control words are applied. The two parameters which impact the reliability of the PUF response are :

1. "Intra-Hamming distance" : The challenge pairs are chosen such that the Hamming Distance HD between them is as great as possible. Ideally HD should be equal to the number delay elements if we consider **complementary** challenges.
2. "Inter-Hamming distance"; The Hamming distance between a challenge pair and any other pair must be as great as possible to avoid correlated

responses.

Considering the parameters values shown on Table 5.2, 63 challenge pairs can be generated with an intra-Hamming distance of 64 (hence complementary challenges) and inter-Hamming distance of 32. This number comes from [BSR] who calculated the sizes of codes $A(n, d, w)$, where n is the number of code bits, d is the minimum distance and w is the constant Hamming Weight. The number of challenge pairs is $n - 1$ when $d = n/2$ and $w = n/2$.

TABLE 5.2 – Challenge pairs parameters.

Parameter name	Default value
Number of delay chains M	1
Number of delay elements per chain N	64
Intra-Hamming distance	64
Inter-Hamming distance	32
Number of challenge pairs (K)	63

5.3.3.2 Estimation of the PUF Measurement Error

This step of the key generation profiling process aims to assess the measurement noise of the loop PUF. To do so, first, we randomly select a challenge pair. Second, we compute the delay difference between the two control words `itnb` times. And finally we deduce the standard deviation of the computed delay differences. We note that it is necessary to indicate the considered measurement window width (`Ww`) since the delay computation and then the obtained standard deviation are highly dependent on it. The higher the `Ww` value, the better the reliability of the measurements (See Figure 2.6). In our case, due to the counter precision limit, the higher value of `Ww` is `0xf`. Over that, we have an overflow and the obtained value is wrong. Table 5.3 indicates the possible and the default values of the needed parameters for the statistical analysis step using our test chips. We describe below the procedure of the preliminary estimation of the PUF measurement error step (Algorithm 1).

TABLE 5.3 – Statistical analysis parameters.

Parameter name	Description	Possible values	Default value
Ww	Width of time window for the loop PUF measurements.	≤0xf at 100MHz	na
itnb	Number of iterations for `stddev` computation.	>0	128

Algorithm 1 Estimation of the PUF measurement error

```
challengeA ← challengeSet[0]
challengeB ← challengeSet[1]
for i = 0 → itnb do
    respA ← LoopPufMeasure (challengeA, 0xf)
    respB ← LoopPufMeasure (challengeB, 0xf)
    diffArray[i] ← respA − respB
end for
stddev ← computeStdDev (diffArray, itnb)
if Ww > 0xf then
    stddev ← stddev << abs (Ww − 0xf)
else if Ww < 0xf then
    stddev ← stddev >> abs (Ww − 0xf)
end if
```

5.3.3.3 Dynamic Reliability Analysis

It is an essential step for the generation of a cryptographic key. Its principle is to repeat the measurement of each bit until it becomes reliable. Hence the reliability of each key bits is enhanced dynamically. Table 5.4 describes the defined parameters to perform the dynamic reliability analysis. If the bit reliability is above a fixed threshold f(TolCoeff, Tmax, Tmin), we regenerate as many times as needed until it becomes reliable or when the regeneration times reaches Tmax.

TABLE 5.4 – Dynamic reliability analysis parameters.

Parameter name	Description	Possible values	Default value
Ww	Width of time window for loop PUF measurements.	≤0xf at 100MHz	na
TolCoeff	Tolerance Coefficient. Its is the most important parameter to define threshold of the bit reliability. The higher is, the most constrainted the reliability procedure is.	> 0	9
Tmin	The minimum number of iterations for reliability bit analysis.	> 0	1
Tmax	The maximum number of iterations for reliability bit analysis. If we reach it, the considered bit is defined as an unreliable bit.	> 0	64
Stddev	The Standard deviation of the measurement error. Already computed at the statistical analysis step.	na	na
KeyLength (K)	Number of challenge pairs.	na	63
ChallengesSet	The list of the challenge pairs.	na	na
lib	List of ignored bits	na	na

Figure 5.9 shows the dynamic reliability analysis procedure for the generation of cryptographic keys. More details are given on Algorithm 2.

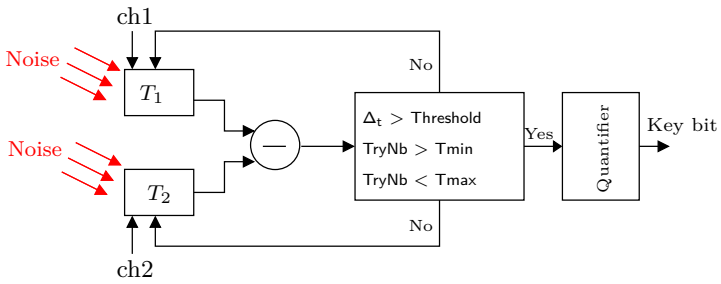


FIGURE 5.9 – Key bit generation with dynamic reliability analysis.

Algorithm 2 Dynamic reliability analysis

```
1: for keybitNb = 0 → KeyLength − 1 do
2:     if keybitNb ∉ lib then
3:         sum ← 0 ;
4:         challengeA ← challengeSet[2 ∗ keybitNb]
5:         challengeB ← challengeSet[2 ∗ keybitNb + 1]
6:         repeat
7:             TryNb ← TryNb + 1
8:             Threshold ← Stddev ∗ TolCoeff ∗ √TryNb
9:             respA ← LoopPufMeasure(challengeA, Ww)
10:            respB ← LoopPufMeasure(challengeB, Ww)
11:            Diff ← respA − respB
12:            Sum ← Sum + Diff
13:        until (TryNb = Tmax) OR ((TryNb > Tmin)AND(abs(sum) > Theshold))
14:        KEY[keybitNb] = Sign(Sum)
15:        if MODE == USER then
16:            DATABANK.Sort(Sum, TryNb, keybitNb)
17:        end if
18:    end if
19: end for
```

For each key bit, we save two main information the sum of delay difference (Sum) and the effective number of tries needed to make the bit steady (TryNb). Note that if the TryNb is equal to Tmax, this means that the bit is still unreliable. The greater the TryNb, the harder the reliability of the bit is. The used reliability threshold depends on the number of tries TryNb, the measurement error Stddev and the confidence level TolCoeff. The used threshold is indeed more constraining when we increase the TolCoef parameter. We note that, at the profiling step, we do not save any information of the bit reliabilities.

5.3.3.4 Reference Key Generation and Unreliable Bit Selection

To generate the reference key, we first generate thousands of samples of the key for a given challenge set. Each one is indeed generated using the dynamic reliability analysis procedure with **higher** Ww value (in our case Ww = 0xf) which leads to a high accuracy on the delay measurements. Due to the measurement

error, all generated keys are not the same (steadiness <100%). Therefore, we propose to deduce the real value of each key bit. for each bit, we first, compute its stability rate (the probability to have '1' on all generated keys). Second, we deduce the bit value (for example, we select '1' if the $\mathsf{prob}(\mathsf{bitvalue} = 1) > 50\%$, 0 otherwise). Using the computed stability rates, we are able to ignore the most unreliable bits. Therefore we propose to generate a list of bit indexes which stability rate is strictly less than a fixed threshold referred to as "bst". Then, according the maximum number of ignored bits (mnib), the worst unreliable bit indexes are listed and their corresponding key bits are set to '0'. We note that when we ignore some bits, the key reliability is enhanced but we reduce the key length and then the key entropy since the ignored bits are always fixed to '0'. Table 5.5 shows the used parameters to generate the reference key after ignoring the most unreliable bits.

TABLE 5.5 – Unreliable bit selection parameters.

Parameter name	Description	Possible values	Default value
mnib	Maximum number of ignored bits.	≥ 0	3
bst	Bit stability threshold.	<1	1

5.3.3.5 Key Code Computation

The key code computation is based on the Hamming code technique. First, we build the parity matrix (Nb_lines, $Nb\_columns$) such that :

- The number of columns ($Nb\_columns$) of the matrix is equal to the keylength.
- The number of lines (Nb_lines) is given by the following formula : $\mathsf{Nb\_lines} = \mathsf{ceil}(\log_2(\mathsf{keyLength}))$.

Each matrix line is indeed a binary mask :

- the first line is a mask which allows us to enable one bit of the key among two.
- the second line is a mask which allows us to enable two bits of the key among four.
- Until the last line, the number of enabled bits is multiplied by two at each line.
- the last line is a mask where all bits are enabled.

An example of the parity matrix (P) for a $\mathsf{keylength} = 63$ is shown below :

$$P = \begin{pmatrix} 10101010101010101010101010101010101010101010101010101010101010101 \\ 11001100110011001100110011001100110011001100110011001100110011001100110 \\ 11110000111100001111000011110000111100001111000011110000111100001111000 \\ 11111111000000001111111100000000111111110000000011111111000000001111111100000000 \\ 11111111111111110000000000000000111111111111111100000000000000001111111111111111 \\ 11111111111111111111111111111111000000000000000000000000000000001111111111111111 \\ 11111111111111111111111111111111111111111111111111111111111111111111111111111111 \end{pmatrix}$$

Second, based on the generated parity matrix, two steps are needed to compute the key code.

1. Each parity matrix line is considered as a binary AND mask and is applied on the given key, the result is stored in a new matrix.
2. In the obtained matrix, we compute each line parity using the XOr operations.

For example, if we consider a key such that
key = "11100101100101111100111111001111010101011011000001111111010010",

after the bitwise operations, we obtain the matrix and then the code shown on Table 5.6.

TABLE 5.6 – Key code computation

Key
11100101100101111100111111001111010101011011000001111111010010

Bitwise AND result	**Key Code bit**
10100000100000101000101010001010000000001010000000010101000000000	**1**
11000100100001001100110011001100010001001000000001001100010001O	**1**
11100000100100001100000011000000010100001011000001110000010000O	**0**
11100101000000001100111100000000010101010000000001111111000000O	**0**
11100101100101111000000000000000001010101101100000000000000000O	**1**
11100101100101111100111111001111000000000000000000000000000000O	**0**
11100101100101111100111111001111010101011011000001111111010010	**0**

The procedure above is applied on the reference key after the ignored bits have been set at '0'. The resulting code is public information from which the original reference key cannot be retrieved. It will be used in two ways.

- First, to detect if the provided key is different from the reference one.
- Second, to help a potential correction system to detect and switch wrong bits.

Hereafter, we present the proposed procedure to generate and correct the key code using the profiling information.

5.3.4 User Key Generation

This step is needed at each cryptographic key generation procedure. It uses the information collected at the profiling step. Using the same challenges used to compute the reference key, we try to generate an identical key to the reference one. Therefore, we define two main steps :

1. Dynamic reliability enhancement : it is the procedure applied to analyse and enhance the reliability of the key. It allows us, first to generate a key that is supposed to be very much similar to the reference key. Despite of the dynamic reliability enhancement efforts, at the end of this step, the obtained key may contain some errors. Therefore, we need to use an error correction algorithm. Second, we save the information about the reliability rate of all key bits in order to ease the key correction process.
2. Key correction : this step allows us to correct the received key depending on the information kept while the previous steps.

Figure 5.10 shows the proposed key generation flow.

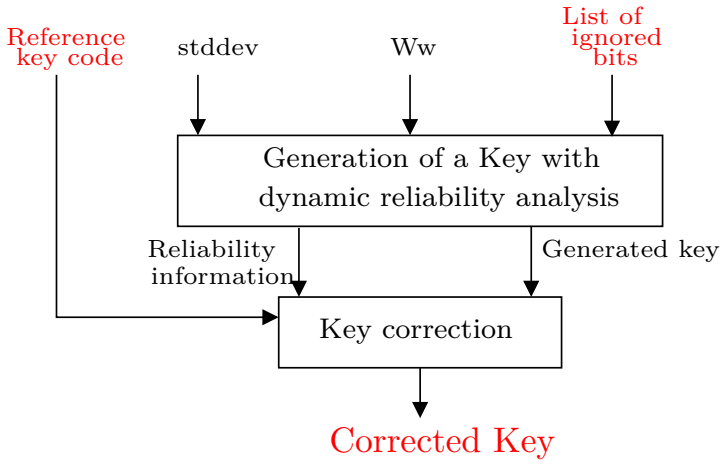


FIGURE 5.10 – Key generation flow.

The key generation with the dynamic reliability analysis is the same procedure used at the profiling stage (Section 5.3.3.3). However, we distinguish two minor variations. First, bits selected to be ignored during the profiling step are not generated, which make this step faster at the user level. Second, the number of performed iterations (lub_it), the final cumulative sum of the delay differences (lub_sum) and the indexes (lub_ind) of the *nub* (number of unreliable bits) worst key bits among the entire key have been stored.

Hereafter we detail the proposed key correction procedure based on the Hamming error correction code. The latter allows us to detect multiple errors however

TABLE 5.7 – Unreliable bit selection parameters.

Parameter name	Description
rkcode	The reference key code.
ukcode	The code of the received key.
key	The key that have to be potentially corrected.
lub_ind	Stored list of bit indexes of most unreliable bits. It is already generated at the dynamic reliability analysis step.
lub_it	List of number of iterations necessary to make key bits of 'lub_ind' reliable.
lub_sum	List of cumulative differences which have been stored during the reliability enhancement step.
nub	Number of unreliable bits that are considered.
mnce	Maximum number of bit errors that we can correct.
lib	List of ignored bits

its correction capacity is limited to one bit. Therefore, we propose to use the Chase algorithm [Cha72] which only considers the list of most unreliable bits. By means of classification all possible error combinations on these bits, a key correction can be effective even in case of multiple errors. The performance of our correction procedure depends on the maximal number of bit errors (*nmce*) that can be corrected. Wrong bits are supposed to be within the *lub_ind* provided list, and the role of our correction procedure is to figure out which one(s). Table 5.7 describes all needed parameters for the key correction step.

Algorithm 3 gives the proposed key correction procedure. First, we have to compute (Section 5.3.3.5) the user key code (ukcode), then if the latter is different from the reference key code (rkcode), the following operations are needed : i) We **enumerate all error combinations** and corresponding syndromes are calculated. ii) We **select the right combination and correct the mistaken bits**. These two steps are detailed below.

Enumeration of error combinations

Based on both, the parity matrix and the information about key reliability, such as the sorted list lub, we propose to compute the possible syndromes. Table 5.8 presents an example of lub with nub = 4.

Considering this list, we first compute all syndromes based on a single bit error. Using the parity matrix, we are able to directly extract the syndromes. The syndrome of a given unreliable bit corresponds to the vector having the same index as the given unreliable bit in the parity matrix.

Algorithm 3 Key correction procedure

```
1: for i = 0 → length(lib) − 1 do
2:     key[lib[i]] =′ 0′;
3: end for
4: ukcode ← getCodeFromKey(key)
5: if rkcode ≠ ukcode then
6:     xorCode = rkcode ⊕ ukcode
7:     listOfPossibleSyndroms ← getSyndroms(mnce, lub_ind, lub_it, lub_sum)
8:     ListMistakenBits ← GetMistakenBits(listOfPossibleSyndroms, lub_ind, xorCode)
9:     correctedKey ← switchMistakenBits(key, ListMistakenBits)
10: end if
```

TABLE 5.8 – Example of an unreliable bits list (lub).

Position in list	**0**	**1**	**2**	**3**
Unreliable bit index	7	45	30	61
Unreliable bit nb. of iterations	64	64	25	12
Unreliable bit sum	16.5	30	125	85

Table 5.9 shows the possible syndromes based on single bit errors for the example shown on 5.8.

TABLE 5.9 – Syndromes based on single bit errors (example).

	7		30		45		61	
1010101	0	10101010101010101010	1	01010101010101	0	10101010101010	1	01
1100110	0	11001100110011001100	0	01100110011001	1	00110011001100	1	10
1111000	0	11110000111100001111100	0	01111000011110	0	00111100001111	0	00
1111111	1	00000000111111111000000	0	01111111100000	0	00111111110000	0	00
1111111	1	11111111000000000000000	0	01111111111111	1	11000000000000	0	00
1111111	1	11111111111111111111111	1	10000000000000	0	00000000000000	0	00
1111111	1	11111111111111111111111	1	11111111111111	1	11111111111111	1	11

However, to handle multiple error situations, we propose to generate all possible combinations using the "binomial coefficient" technique. The number of possible errors combination which can be considered is given by

$$\mathsf{Nb}_{\mathsf{comb}} = \sum_{\mathsf{i}=1}^{\mathsf{mnce}} \binom{\mathsf{nub}}{\mathsf{i}}. \tag{5.2}$$

For each error combination, the corresponding syndrome is computed Xoring the syndromes of the unreliable bits composing the combination. Table 5.10 shows an example of a selection of possible syndromes based on the example given in Table 5.8 with a $\mathsf{mnce} = 3$ and $\mathsf{nub} = 4$.

TABLE 5.10 – A selection of possible syndromes (example).

Combinations	Number of errors	Computations	Syndrome
(7)	1	$syndrom = c(7)$	"0001111"
(7 , 30)	2	$syndrom = c(7) \oplus c(30)$	"1001100"
(30 , 45 , 61)	3	$syndrom = c(30) \oplus c(45) \oplus c(61)$	"1000111"

In order to save the reliability of each error combination, the sum of all needed iterations with the mean of the cumulative sum of delay differences for each error combination is saved (see example given in Table 5.11).

TABLE 5.11 – Stored data for each error combination (example).

Combinations	Sum of tries number	Cumulative sum
(7)	64	16.5
(7, 30)	64 + 25	16.5 + 125
(30, 45, 61)	25 + 64 + 12	125 + 33 + 8

After generating all the possible syndromes, hereafter we present how to select and then correct the mistaken bits in the received bad key.

Selection of the Right Combination and Key Correction

Based on all the information collected in the previous steps, we distinguish three possible cases to select and correct the mistaken bits if it is possible. The mistaken bits are selected such that the corresponding syndrome is equal to the reference syndrome. The latter is indeed obtained when Xoring the reference key code and the received key code ($\mathsf{Ref\_syndrome} = \mathsf{rkcode} \oplus \mathsf{ukcode}$).

1. Among all the possible syndromes, **only one** corresponds to the Ref_syndrome. In this case, the corresponding error combination is selected, and then the involved bits are detected and switched on the received key. After that, the received key will be correct.
2. Among all possible error combinations, more than one error combinations have the same syndrome which is equal to the reference one. In this case, among these error combinations, we have to choose (to select) the one that have the higher probability to be the right one (which allows us to correct the received key). Based on the "Chase" algorithm principle, we propose to select the best error combination based on its reliability rate (weight).

   All the error combinations that have equivalent syndromes are sorted :

   (a) By decreasing order of the number of involved mistaken bits.

   (b) Then by decreasing order of the number of tries.

(c) Finally by increasing order of the cumulative sum.

Table 5.12 shows an example of the adopted procedure to select the good error combination in case of equivalent syndromes. After selecting the error combination, we switch the involved bits on the received key to propose a possible correction.

3. Among all possible error combinations, there are no syndromes equal to the reference one. This means that either the error affect one or multiple bits which are considered more reliable than those already selected or the error combination is composed of a number of errors greater than the fixed mnce parameter. In this case, the correction of the received key remains not possible.

TABLE 5.12 – Error combination selection procedure (case : multiple equivalent syndromes).

Combination	**Sum of number of tries**	**Global sum**	Choice
(1 , 13, 62)	**192**	**13**	**Selected**
(7 , 9 , 11)	192	55	
(4 , 8 , 9 , 18 , 62)	76	250	

In this section, we detailed the proposed procedure to generate and correct a cryptographic key using the loop PUF structure. Hereafter we propose to study the influence of each parameter on the generated key quality in terms of time consumption, key length and post-processing block complexity. Therefore, in the next section , we present and discuss the elaborated statistics on the key generated quality (Binary Error Rate BER and the Key Error rate KER) when varying the different parameters.

5.3.5 Experimental Results and Discussions

In this section, we are interested in evaluating the proposed cryptographic key generation procedure. The latter can be evaluated according to three main characteristics.

1. The error rate.
2. The key generation time consumption.
3. The Key length.

To evaluate them, experiments have been carried out on 2 ASICs, each one embeds 49 loop PUFs. Hereafter we propose, first, to investigate the influence of the parameters on the three characteristics. Second, we show and discuss some experimental results. And finally, we present a method that allows the user to select the parameter values based on their needs in terms of binary error rate, key generation time, and the key length.

5.3.5.1 The Influence of the Parameters on the Characteristics

As seen at the beginning of Section 5.3, in order to generate a cryptographic key, several parameters are involved. Some of them have a direct and a high influence on the characteristics of the key generation procedure. Table 5.13 shows the degree of influence of the main parameters on the characteristics of the proposed key generation procedure.

TABLE 5.13 – The influence of some parameters on the key generation characteristics.

Parameter	Time Consumption	Key Length	Error Rate
Ww ↗	↑↑	-	↓↓
Tmin ↗	↑	-	↓
Tmax ↗	↑	-	↓
TolCoef ↗	↑	-	↓
nub ↗	-	-	↓
mnce ↗	-	-	↓↓
mnib ↗	↓↓	↓↓	↓↓

Hereafter, we detail the main dependency between each characteristic and the parameter.

Error Rate

The error rate characteristic shows the performance of the key generation procedure in terms of reliability. In information theory, it is considered as the main characteristic of key correction procedures. It indeed indicates the ability of the key generation procedure to produce a key identical to the reference one. Table 5.13 shows that all the involved parameters have an impact on the error rate characteristic. However, there are three main parameters that have a higher influence on it which are Ww, mnce and mnib. In the next section, we present some experimental results showing the impact of the involved parameters on the BER characteristic when varying them.

The Key Generation Time Consumption

Among all the involved parameters during the key generation procedure, the time consumption is highly influenced by both the window width (Ww) and the maximum number of ignored bits (mnib) parameters. Both of them impact directly the key generation time at the dynamic reliability analysis step. When we increase the Ww, this means that we need much more time to get the PUF response for the applied control word. However, when we increase the mnib value, we reduce considerably the key generation time since the most unreliable bits are ignored and then, their reliabilities are not analysed. Figure 5.11 shows the evolution of

the key generation time when varying the measurement window width (Ww) and the mnib parameter from 0 to 10.

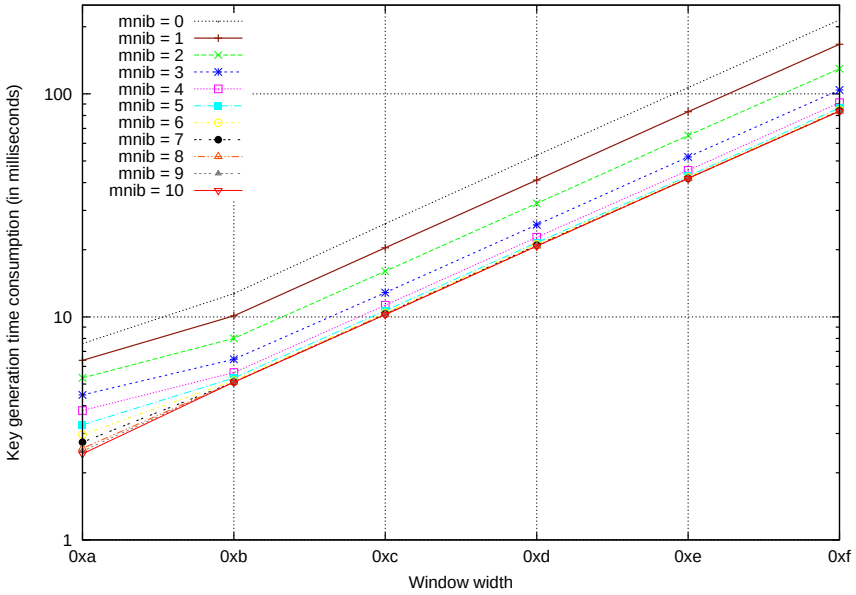


FIGURE 5.11 – Impact of mnib and the Ww parameters on the key generation time.

Key Length

The PUF key and its reliability are highly dependent from both the challenge set introduced and the PUF itself due to the uniqueness property of the PUF. Based on the same challenge set, the number of unreliable bits varies from a loop PUF to another. In order to illustrate the disparity of the loop PUFs quality in terms of number of unreliable bits, two chips embedding 49 PUF samples have been evaluated.

For each PUF, the number of unstable bits has been computed, and then cartography (Figure 5.12) has been drawn. We note that the PUF quality is homogeneously scattered on the ASIC, regardless of its physical location. Figure 5.13 shows the distribution of the number of unstable bits on the 98 studied PUFs.

We note that 10% of the studied PUFs present a perfect quality (zero unreliable bits) while 80% have less than 5 unreliable bit. Only 10% include 5 or 6 unreliable bits. To counterbalance the PUF quality disparity, we propose to ignore some unreliable bits which reduce the key length in order to enhance the key error rate.

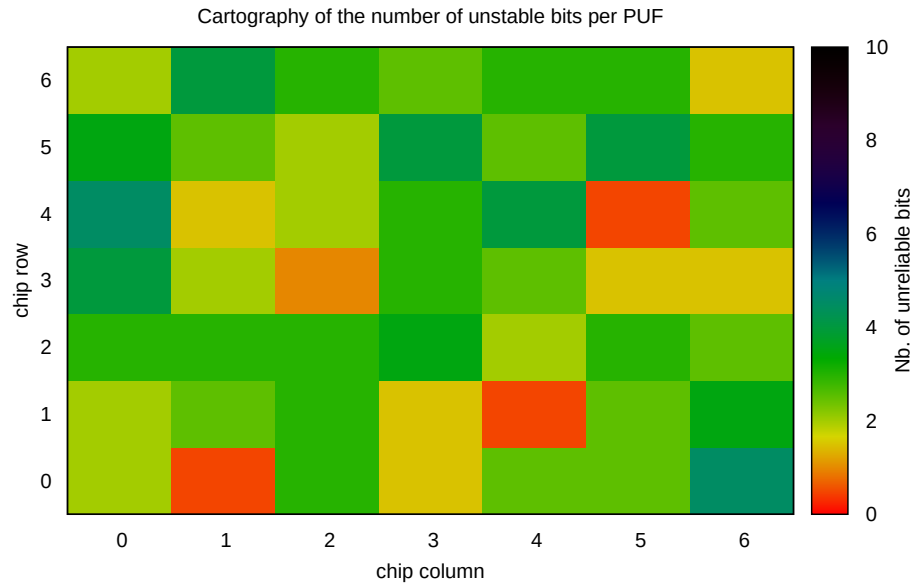


FIGURE 5.12 – Cartography of unstable bits on two chips

Hereafter we show some experimental results that better explain the dependency between the parameters and the performance of the key generation procedure.

5.3.5.2 Experimental Results

In this section, we propose to evaluate the evolution of the BER and the needed generation time when varying the key length. Below, three scenarios are explored. We note that there is no influence of the correction procedure on the generation time consumption.

Error Rate Evaluation Without Correction Scheme ($53 \leq \mathsf{KeyLength} \leq 63$)

In this scenario, we suppose that the user wants a lightweight key generation system and has not a hard constraint about key length. Figure 5.14 shows that when varying the mnib parameter, we are able to reduce both the BER and the key generation time. We note that we reach a BER $\leq 10^{-9}$, without error correction schemes which means without extra logic, when ignoring at maximum 6 bits and

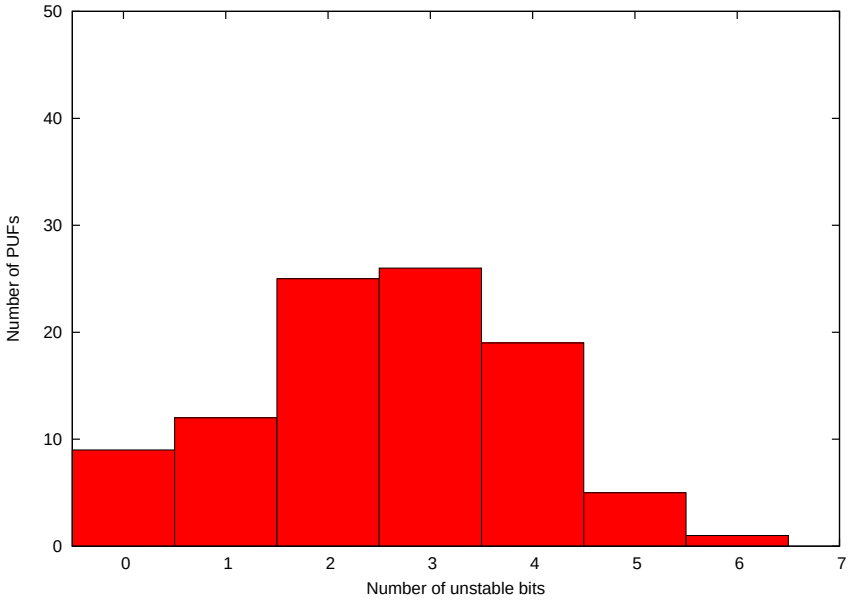


FIGURE 5.13 – Histogram of the number of unstable bits in 98 PUF samples.

then producing a key with 57 bits in 40 ms. Moreover a key can be generated in 10 ms with a BER $\leq 10^{-9}$ when $\mathsf{mnib} \geq 7$.

Error Rate Evaluation with a Correcting Scheme ($\mathsf{KeyLength} = 63$)
In this scenario, we suppose that the user has a hard constraint about key length and wants to use his key generation system at its full abilities. To do so, all challenge pairs are considered and our Hamming/Chase corrector is used. The corrector is parameterized with $\mathsf{mnce} = 5$ and $\mathsf{nub} = 5$ which means that 5 wrong bits can be corrected at maximum. The impact of our corrector is undeniable, comparing to results obtained without correcting scheme, the BER is divided by 100. However, experimental results given by Figure 5.15 ($\mathsf{mnib} = 0$) shows that the best obtained BER is greater than 10^{-5} which could be considered as not sufficient in a such critical application.

Error Rate Evaluation with a Correcting Scheme, Preserving a High Key Length ($60 \leq \mathsf{KeyLength} \leq 63$)
In this scenario, the objective is to reach the best BER. Therefore we propose to use our correcting scheme ($\mathsf{mnce} = 5$ and $\mathsf{nub} = 5$) and to reduce the key length by ignoring a few unreliable bits. Experimental results (Figure 5.15) show that we are able to reach a BER lower than 10^{-9} when ignoring only two bits when

measuring the delays at the highest window value of Ww = 0xf.

We also study the BER when varying the mnce parameter from 1 to 5. According to the results shown on the Figures 5.16a, 5.16b, 5.17a and 5.17b, we can see that the impact of mnce is high.

In addition to the three main characteristic, we note that the complexity of the proposed module is important. Hereafter we propose to study the hardware complexity of the proposed key generation module using the loop PUF structure.

Hardware Implementation Complexity

The reliability enhancement method has been validated by software driving the 49 PUF cores of ASIC prototypes. In order to compare our results with previous ones, the implementation complexity of the proposed method has been studied in Virtex 5 FPGA. The architecture includes the dynamic reliability analysis procedure, and has been synthesized either with or without error correction. The results show that the error correction greatly increases the complexity with a very limited gain in terms of key bits. Without any correction, the complexity remains very low (117 slices) but still provides a high level of reliability as the BER is less than 10^{-9}. The measurement time of $10ms$ is not negligible but could be acceptable in most applications.

TABLE 5.14 – Hardware complexity of the error correction algorithm : number of occupied slices in Xilinx Virtex 5 technology.

Loop PUF complexity	20	
adaptive key quantification	97	
Key correction complexity	0	235
Total complexity	117	352
BER at 10 ms	10^{-9}	10^{-5}
BER at 100 ms	10^{-9}	10^{-9}
key length	≥ 56	≥ 61

In the next section we present large scale results of the key generation characteristics. We discuss how the user can find a trade-off between the different involved parameters in order to satisfy its needs.

Making a Trade-off Between Parameters to Achieve Specific Constraints

As discussed above, the BER evolution depends on different parameters. And it is also influenced by the three other characteristics that characterize our key generation procedure.

Table 5.15 shows a large scale characterization of the key generation procedure. We present the BER results when :

- The maximum key length vary from 60 to 63 bits ($\mathsf{mnib} \in [0, 3]$).
- There is no correction schemes when ignoring until 10 bits ($\mathsf{mnce} = 0$ and $\mathsf{mnib} \in [0, 10]$).
- The maximum considered error numbers is between 1 bit and 5 bits ($\mathsf{mnce} \in [1, 5]$) with an unreliable bit number (nub) of 5.
- The measurement window width is between 0xa and 0xf ($\mathsf{Ww\ in\ (0xa, 0xb, 0xc, 0xd, 0xe, 0xf)}$).

5.4 Conclusion

In this chapter, we proposed two applications of the proposed loop PUF, device authentication and cryptographic key procedures. For the device authentication purposes, we propose a method based on the Pearson coefficient. It allows us to authenticate a device on a reliable way as shown by the experimental results performed on 18 ASIC platforms. We concluded that using the loop PUF structure, and due to the accuracy of its output and the method that we propose, we are able to distinguish a PUF from other similar PUFs either implemented in the same ASIC or placed at the same place in different ASICs without post processing schemes and with a large error margin, even when varying the temperature. The proposed method is not time consuming and it takes into account the scaling phenomenon caused by the temperature variation.

We also presented a method to make a PUF-based cryptographic key generator very reliable and low-cost. It benefits from the Loop PUF which allows the key generator to exploit a huge set of challenges, and a specific profiling stage. The BER has been measured for many configurations, like the measurement time, the unreliable bits to ignore, the ECC performances. The latter is based on a low complexity Hamming/Chase algorithm which considers the most unreliable bits. The results show very good performances as $BER < 10^{-9}$ with or without ECC to generate 61 bits of keys. This is mainly due to the accurate profiling stage which takes advantage of many actions to improve the reliability : use distant challenges, extract unreliable bits, increase the number of measurements for unreliable bits, add correction codes. It is interesting to notice that the key can be generated without any error correction schemes by reducing slightly the number of bits, thus leading to a very lightweight PUF which needs only 117 slices in a Virtex5 FPGA.

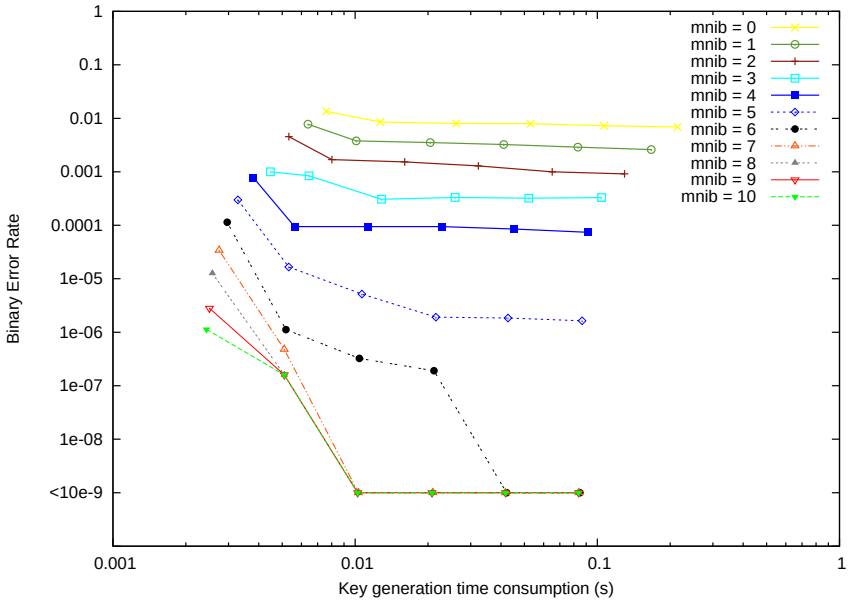


FIGURE 5.14 – The BER evolution without correction schemes when varying the mnib parameter.

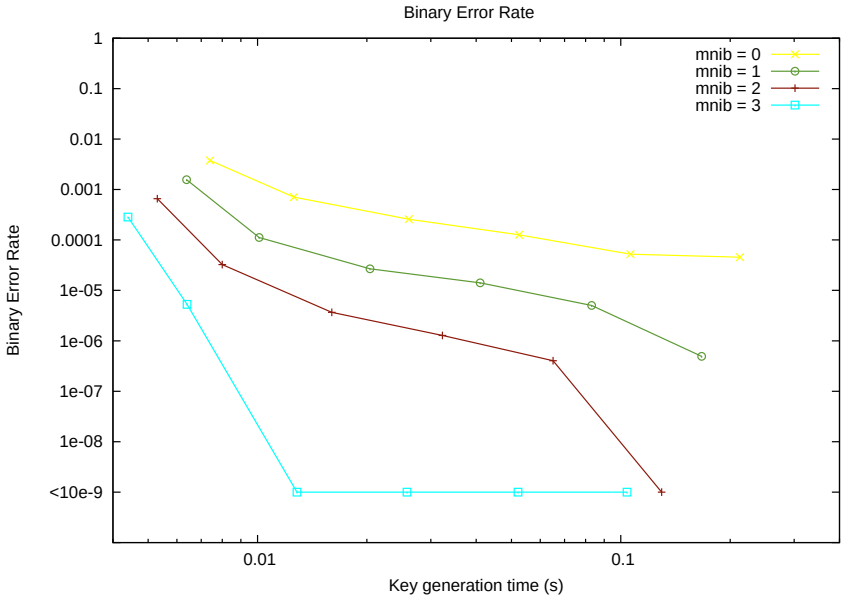
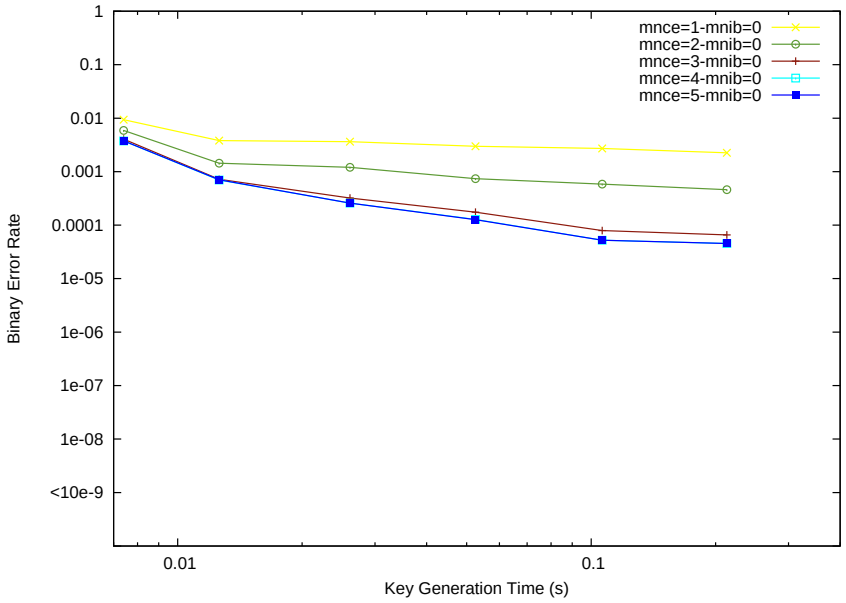
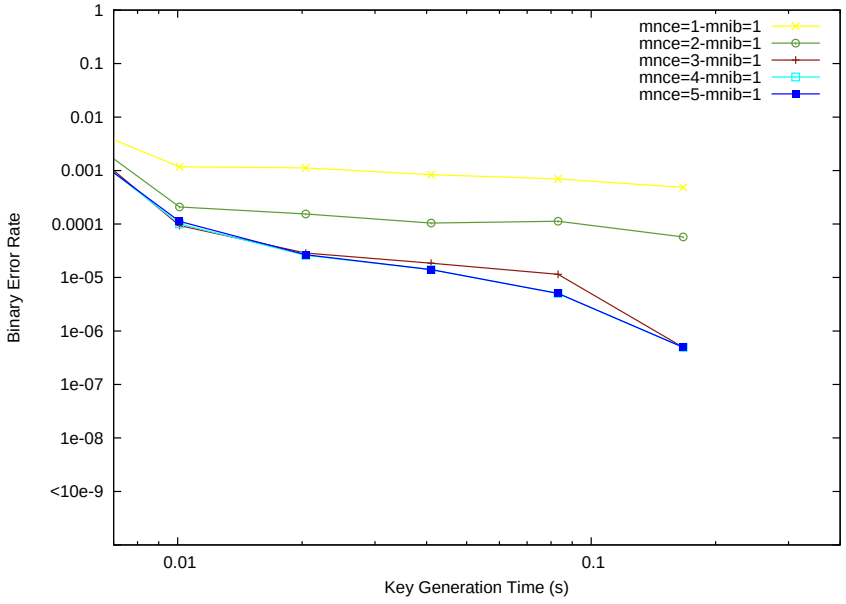


FIGURE 5.15 – The BER evolution when varying the key length using a correction scheme.

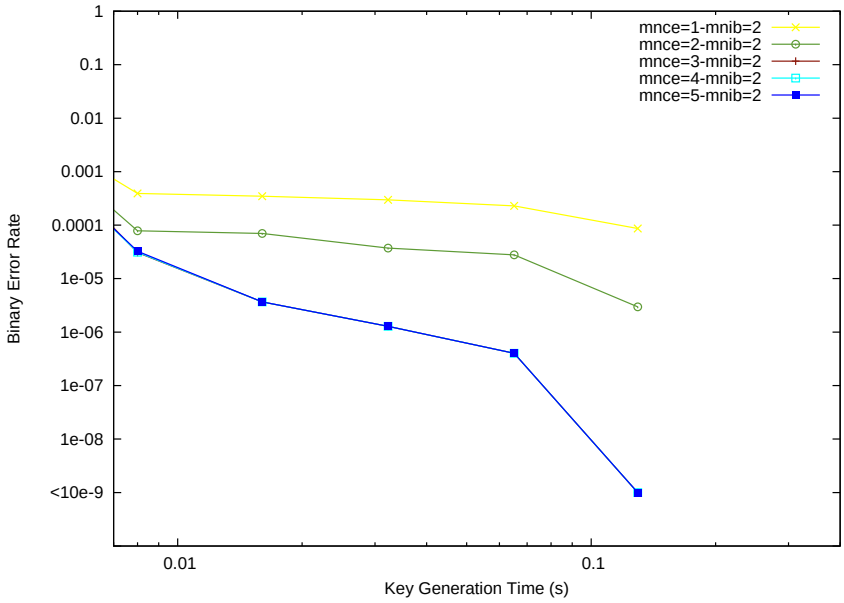


(a) The mnib = 0 (keyLength = 63).

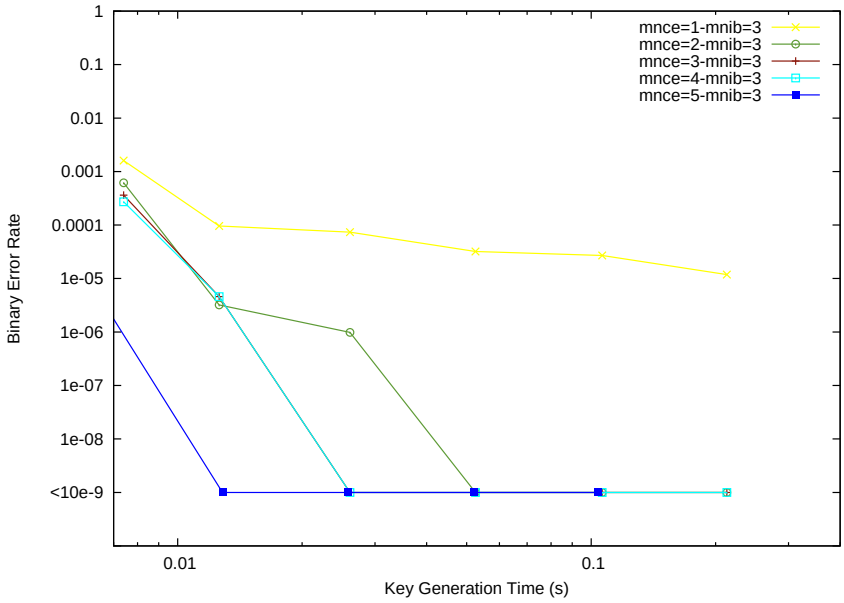


(b) The mnib = 1 (keyLength = 62).

FIGURE 5.16 – The BER evolution when varying the key length (mnib=0 and mnib=1).



(a) The mnib = 2 (keyLength = 61).



(b) The mnib = 3 (keyLength = 60).

FIGURE 5.17 – The BER evolution when varying the key length (mnib=2 and mnib=3).

TABLE 5.15 – How to set the parameters value knowing our needs.

		Ww/time(ms)					
mnce	mnib/KeyLength(bits)	0xa/7	0xb/13	0xc/26	0xd/52	0xe/106	0xf/213
0	0/63	1.35e-02	8.54e-03	8.02e-03	7.94e-03	7.26e-03	6.91e-03
	1/62	7.77e-03	3.78e-03	3.52e-03	3.24e-03	2.89e-03	2.61e-03
	2/61	4.54e-03	1.69e-03	1.53e-03	1.29e-03	1.00e-03	9.14e-04
	3/60	1.00e-03	8.42e-04	3.06e-04	3.32e-04	3.22e-04	3.29e-04
	4/59	9.43e-05	9.43e-05	9.43e-05	9.43e-05	8.53e-05	7.40e-05
	5/58	2.98e-04	1.66e-05	5.18e-06	1.91e-06	1.91e-06	1.64e-06
	6/57	1.14e-04	1.12e-06	3.24e-07	1.91e-07	1.00e-09	1.00e-09
	7/56	3.40e-05	4.79e-07	1.00e-09	1.00e-09	1.00e-09	1.00e-09
	8/55	1.27e-05	1.60e-07	1.00e-09	1.00e-09	1.00e-09	1.00e-09
	9/54	2.82e-06	1.60e-07	1.00e-09	1.00e-09	1.00e-09	1.00e-09
	10/53	1.13e-06	1.60e-07	1.00e-09	1.00e-09	1.00e-09	1.00e-09
1	0/63	1.01e-02	4.50e-03	4.14e-03	3.23e-03	2.72e-03	2.52e-03
	1/62	5.40e-03	1.56e-03	1.34e-03	8.28e-04	6.71e-04	6.04e-04
	2/61	2.60e-03	2.81e-04	1.58e-04	1.35e-04	1.33e-04	9.72e-05
	3/60	1.43e-03	1.02e-04	3.36e-05	1.13e-05	3.04e-05	9.72e-06
2	0/63	6.75e-03	1.74e-03	1.30e-03	7.25e-04	6.27e-04	5.39e-04
	1/62	2.90e-03	2.77e-04	1.09e-04	8.95e-05	5.79e-05	7.01e-05
	2/61	1.21e-03	1.86e-05	3.24e-06	3.24e-06	8.10e-06	2.43e-06
	3/60	5.41e-04	2.43e-06	8.10e-07	1.00e-09	1.00e-09	1.00e-09
3	0/63	4.67e-03	8.07e-04	3.44e-04	1.14e-04	8.60e-05	7.82e-05
	1/62	2.02e-03	1.36e-04	4.05e-05	3.64e-06	2.02e-06	2.83e-06
	2/61	8.88e-04	2.02e-05	1.00e-09	1.00e-09	1.00e-09	1.00e-09
	3/60	3.55e-04	5.67e-06	1.00e-09	1.00e-09	1.00e-09	1.00e-09
4	0/63	4.32e-03	7.94e-04	3.22e-04	7.94e-05	6.46e-05	4.37e-05
	1/62	1.85e-03	1.38e-04	4.09e-05	2.83e-06	2.02e-06	3.64e-06
	2/61	7.50e-04	2.31e-05	1.00e-09	1.00e-09	1.00e-09	1.00e-09
	3/60	2.50e-04	5.67e-06	1.00e-09	1.00e-09	1.00e-09	1.00e-09
5	0/63	4.36e-03	8.08e-04	3.24e-04	8.06e-05	6.46e-05	4.37e-05
	1/62	1.86e-03	1.47e-04	4.21e-05	2.83e-06	2.02e-06	3.64e-06
	2/61	7.17e-04	3.16e-05	1.00e-09	1.00e-09	1.00e-09	1.00e-09
	3/60	2.60e-04	6.88e-06	1.00e-09	1.00e-09	1.00e-09	1.00e-09

Chapitre 6

Conclusion and Perspectives

In this chapter, we give general concluding remarks and present directions for future research.

6.1 Conclusions

Physically Unclonable Functions or PUFs are emergent physical security primitives. They are able to secure the devices against counterfeit problems, to protect cryptographic key generation and storage against physical and mathematical attacks, etc. In this thesis, we focused on silicon PUF constructions, properties, characterization methods and their applications as a solution to face security issues.

In order to facilitate the design implementation and to enhance the security level of existing PUFs, we have proposed two novel silicon PUF structures. Our first contribution was dedicated to the description and the evaluation of a delay-based PUF structure called "loop PUF". We showed that the proposed architecture is easy to design either on FPGA or on ASIC platforms. Moreover, due to the non differential structure, we have a high number of possible challenges. Also this PUF is appropriate to devise methods aiming at enhancing the reliability of the response. The proposed structure has been evaluated using a statistical computation metrics, and designed with a 65nm technology on both FPGA and ASIC platforms. Practical experiments showed that the proposed structure presents interesting performances. The second proposed PUF structure, called "TERO PUF", is a Transient Effect Ring Oscillator based PUF. We showed that the main benefit of this structure is that, theoretically, it is not sensitive to locking phenomenon. The implementation process and the experimental results of this construction on an ALTERA FPGA platform are described in detail in this manuscript. We concluded that the TERO PUF contains interesting uniqueness and steadiness performances.

As a third contribution, we proposed a novel method to characterize the delay PUF performances. It takes advantage from measurements of the physical values. The main contribution of this method is that, unlike the classical methods, it does not require a large number of tries to evaluate the PUF performance. Moreover, it can be applied at design stage. We identified the most important properties that a PUF has to meet and then proposed the corresponding metrics. We also validated the proposed metrics by comparing the characterization results of two delay PUF structures using our method and existing ones based on statistical tests.

Next, we proposed two loop PUF-based security applications. We first studied how to use the loop PUF response to authenticate an integrated circuit (IC). We proposed a method based on the measurement of physical values of the loop PUF basic delay elements and using the Pearson coefficient as an authentication metric. Based on this scheme, the authentication performance and the efficiency of the loop PUF is evaluated. Second, we proposed a new loop PUF-based cryptographic key generation method. We took advantage of many existing techniques to enhance the reliability of unsteady data. The experimental results showed the

efficiency of the proposed loop PUF-based key generation method.

6.2 Future Research

Several interesting areas of future research arise from this work. In what follows, we point out some of these research directions.

An immediate perspective of our first contribution would be to carry out accurate analyzes of the robustness against modeling and physical attacks. It would be thereafter interesting to enhance our proposed loop PUF structure by proposing a lightweight security solution as a countermeasure for the possible attacks. One useful future work from a practical point of view would be to describe the algorithmic approach for the challenge generation process based on our theoretical research [CDG+13].

Future research on the work of the TERO PUF structure would be to evaluate its robustness against Electro-Magnetic attacks in order to validate our theoretical assumptions. A further perspective would be to optimize the implementation area of the TERO PUF to improve its design complexity. It would be also helpful to study the use of the least significant bits of the TERO loop counter to generate random numbers. The proposed TERO PUF cell will thus simultaneously serve as a TRNG and as a PUF functional block.

An immediate perspective of the proposed characterization method is to confirm that it provides the same results when using simulation model and physical implementation, for the evaluation of a given PUF structure. One important extension, from a theoretical point of view, is to further extend the proposed metrics for other PUF constructions, for instance RO PUFs.

List of Publications

[A] Zouha Cherif, Florent Flament, Jean-Luc Danger, Shivam Bhasin, Sylvain Guilley and Hervé Chabanne, **Evaluation of White-Box and Grey-Box Noekeon Implementations in FPGA**, Reconfigurable Computing and FPGAs (ReConFig), 2010 International Conference on , vol., no., pp.310,315, 13-15 Dec. 2010 doi : 10.1109/ReConFig.2010.36

[B] Zouha Cherif, Lilian Bossuet and Jean-Luc Danger, **Performance Evaluation of Silicon Physically Unclonable Function by Studing Physicals Values**, New Circuits and Systems Conference (NEWCAS), 2011 IEEE 9th International , vol., no., pp.482,485, 26-29 June 2011 doi : 10.1109/NEWCAS.2011.5981324

[C] Zouha Cherif, Jean-Luc Danger, Sylvain Guilley and Lilian Bossuet, **An Easy-to-Design PUF Based on a Single Oscillator : The Loop PUF**, Digital System Design (DSD), 2012 15th Euromicro Conference on , vol., no., pp.156,162, 5-8 Sept. 2012 doi : 10.1109/DSD.2012.22

[D] Lubos Gaspar, Viktor Fischer, Tim Guneysu and Zouha Cherif, **Two IP Protection Schemes for Multi-FPGA Systems**, Reconfigurable Computing and FPGAs (ReConFig), 2012 International Conference on , vol., no., pp.1,6, 5-7 Dec. 2012 doi : 10.1109/ReConFig.2012.6416790

[E] Zouha Cherif, Jean-Luc Danger, Sylvain Guilley, Jon-Lark Kim and Patrick Solé, **Multiply Constant Weight Codes**, Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on , vol., no., pp.306,310, 7-12 July 2013 doi : 10.1109/ISIT.2013.6620237

[F] Zouha Cherif, Jean-Luc Danger, Florent Lozach́, Yves Mathieu, Lilian Bossuet and Tarik Graba, **Evaluation of Delay PUFs on CMOS 65nm Technology : ASIC vs FPGA**, In Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy (HASP ́13). ACM, New York, NY, USA, , Article 4 , 8 pages. DOI=10.1145/2487726.2487730

[G] Lilian Bossuet, Xuan Ngo, Zouha Cherif and Viktor Fischer, **A PUF Based on Transient Effect Ring Oscillator and Insensitive to Locking Phenomenon**, To appear in IEEE Transactions on Emerging Topics in Computing journal on 2013.

[H] Zouha Cherif, Jean-Luc Danger, Sylvain Guilley, Han Mao Kiah, Jon-Lark Kim, Chee Yeow Meng, Patrick Solé and Zhang Xiande **Multiply Constant-Weight Codes and the Reliability of Loop Physically Unclonable Functions**, Submitted to the IEEE Transactions on Information Theory Journal (Under Review).

[I] Zouha Cherif, Florent Lozac'h, Jean-Luc Danger and Lilian Bossuet. **Lightweight PUF-Based Key Generation with Great Reliability**, Submitted to the IEEE International Symposium on Hardware-Oriented Security and Trust (HOST'2014).

Bibliographie

[ALT] ALTERA, *Cyclone-II device Handbook-Best Practices for Incremental Compilation Partitions and Floorplan Assignments.*

[BBAF13] P. Bayon, L. Bossuet, A. Aubert, and V. Fischer, *Electromagnetic analysis on ring oscillator-based true random number generators*, Circuits and Systems (ISCAS), 2013 IEEE International Symposium on, 2013, pp. 1954–1957.

[BCI09] J. Bringer, H. Chabanne, and T. Icart, *On physical obfuscation of cryptographic algorithms*, Proceedings of the 10th International Conference on Cryptology in India : Progress in Cryptology (Berlin, Heidelberg), INDOCRYPT '09, Springer-Verlag, 2009, pp. 88–103.

[BHP11] C. Bohm, M. Hofer, and W. Pribyl, *A microcontroller sram-puf*, Network and System Security (NSS), 2011 5th International Conference on, sept. 2011, pp. 269 –273.

[BSR] A. Brouwer, N. Sloane, and E.M. Rains, *Constant weight codes*, http ://www.win.tue.nl/ aeb/codes/Andw.html.

[CDG+13] Z. Cherif, J-L. Danger, S. Guilley, J-L. Kim, and P. Sole, *Multiply constant weight codes*, Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on, 2013, pp. 306–310.

[CDGB12] Z. Cherif, J-L. Danger, S. Guilley, and L. Bossuet, *An easy to design puf based on a single oscillator : the loop puf*, DSD'12, 2012.

[Cha72] D. Chase, *Class of algorithms for decoding block codes with channel measurement information*, Information Theory, IEEE Transactions on **18** (1972), no. 1, 170–182.

[DK07] G. DeJean and D. Kirovski, *Rf-dna : Radio-frequency certificates of authenticity*, Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings, Lecture Notes in Computer Science, vol. 4727, Springer, 2007, pp. 346–363.

[DORS08] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith, *Fuzzy Extractors : How to Generate Strong Keys from Biometrics and Other Noisy Data*, SIAM J. Comput. **38** (2008), no. 1, 97–139.

[DRS04] Y. Dodies, L. Reyzin, and A. Smith, *Fuzzy extractors : How to generate strong keys from biometrics and other noisy data*, EUROCRYPT, 2004, pp. 523–540.

[FIP01] FIPS, *Security requirements for cryptographic modules*, May 25 2001, Publication 140-2. http ://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf.

[Gas03] B. Gassend, *Physical Random Functions*, 2003, Msc thesis, MIT.

[GCvDD02] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, *Silicon physical random functions*, ACM Conference on Computer and Communications Security, 2002, pp. 148–160.

[GKJST07] J. Guajardo, S. Kumar, G. Jan Schrijen, and P. Tuyls, *FPGA Intrinsic PUFs and Their Use for IP Protection*, CHES (P. Paillier and I. Verbauwhede, eds.), Lecture Notes in Computer Science, vol. 4727, Springer, 2007, pp. 63–80.

[GLC+04] B. Gassend, D. Lim, D. Clarke, S. Devadas, and Marten van Dijk, *Identification and authentication of integrated circuits*, Concurrency and Computation : Practice and Experience **16** (2004), no. 11, 1077–1098.

[GvT+09] J. Guajardo, B. Škorić, P. Tuyls, S. Kumar, T. Bel, Blom A., and G. J. Schrijen, *Anti-counterfeiting, key distribution, and key storage in an ambient world via physical unclonable functions*, Information Systems Frontiers **11** (2009), no. 1, 19–41.

[HBF07] D. Holcomb, W.P. Burleson, and K. Fu, *Initial sram state as a fingerprint and source of true random numbers for rfid tags*, In Proceedings of the Conference on RFID Security, 2007.

[HBF09] D.E. Holcomb, W.P. Burleson, and K. Fu, *Power-up sram state as an identifying fingerprint and source of true random numbers*, Computers, IEEE Transactions on **58** (2009), no. 9, 1198 –1210.

[HBNS13] C. Helfmeier, C. Boit, D. Nedospasov, and J-P. Seifert, *Cloning physically unclonable functions*, Hardware-Oriented Security and Trust (HOST), 2013 IEEE International Symposium on, 2013, pp. 1–6.

[HST10] Helena Handschuh, Geert-Jan Schrijen, and Pim Tuyls, *Hardware intrinsic security from physically unclonable functions*, Towards Hardware-Intrinsic Security (A.R. Sadeghi and D. Naccache, eds.), Information Security and Cryptography, Springer Berlin Heidelberg, 2010, pp. 39–53 (English).

[HYKS10] Y. Hori, T. Yoshida, T. Katashita, and A. Satoh, *Quantitative and statistical performance evaluation of arbiter physical unclonable functions on fpgas*, Reconfigurable Computing and FPGAs, International Conference on **0** (2010), 298–303.

[II] Intrinsic-ID, *Saturnus, bring-your-own-security (byos) for your data in the cloud. secure access and sharing of data. anytime, anywhere and on any device*, http ://www.intrinsic-id.com/products/saturnus-/sthash.VMSkj3OH.dpuf.

[KGM+08] S.S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls, *Extended abstract : The butterfly puf protecting ip on every fpga*, Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on, june 2008, pp. 67 –70.

[KJJ99] Paul C. Kocher, J. Jaffe, and B. Jun, *Differential power analysis*, Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology (London, UK, UK), CRYPTO '99, Springer-Verlag, 1999, pp. 388–397.

[KKR+12] S. Katzenbeisser, Ü. Kocabaş, V. Rožić, A-R Sadeghi, I. Verbauwhede, and C. Wachsmann, *Pufs : Myth, fact or busted ? a security evaluation of physically unclonable functions (pufs) cast in silicon*, Cryptographic Hardware and Embedded Systems – CHES 2012 (Emmanuel Prouff and Patrick Schaumont, eds.), Lecture Notes in Computer Science, vol. 7428, Springer Berlin Heidelberg, 2012.

[KS11] W. Killmann and W. Schindler, *A proposal for : Functionality classes for random number generators1*, September 2011.

[LDT00] K. Lofstrom, W.R. Daasch, and D. Taylor, *Ic identification circuit using device mismatch*, Solid-State Circuits Conference, 2000. Digest of Technical Papers. ISSCC. 2000 IEEE International, 2000, pp. 372–373.

[LHA+12] Arjen K. Lenstra, James P. Hughes, Maxime Augier, Joppe W. Bos, Thorsten Kleinjung, and Christophe Wachter, *Ron was wrong, whit is right.*, IACR Cryptology ePrint Archive **2012** (2012), 64, informal publication.

[Lim04] Lim, D., *Extracting Secret Keys from Integrated Circuits*, Ph.D. thesis, MIT, 2004.

[LLG+04] J.W. Lee, D. Lim, B. Gassend, G.E. Suh, M. van Dijk, and S. Devadas, *A technique to build a secret key in integrated circuits for identification and authentication applications*, VLSI Circuits, 2004. Digest of Technical Papers. 2004 Symposium on, june 2004, pp. 176 – 179.

[Mae12] Maes, R. , *Physically Unclonable Functions : Constructions, Properties and Applications*, Ph.D. thesis, KU Leuven, 2012.

[MCMS10] A. Maiti, J. Casarona, L. McHale, and P. Schaumont, *A large scale characterization of ro-puf*, HOST'10, 2010, pp. 94–99.

[MKD10] M. Majzoobi, F. Koushanfar, and S. Devadas, *Fpga puf using programmable delay lines*, 2010, pp. 1 –6.

[MS09] A. Maiti and P. Schaumont, *Improving the quality of a physical unclonable function using configurable ring oscillators*, Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on, 31 2009-sept. 2 2009, pp. 703 –707.

[MS11] ———, *Improved ring oscillator puf : An fpga-friendly secure primitive*, J. Cryptol. **24** (2011), no. 2, 375–397.

[MSSS11] D. Merli, D. Schuster, F. Stumpf, and G. Sigl, *Semi-invasive em attack on fpga ro pufs and countermeasures*, Proceedings of the Workshop on Embedded Systems Security (New York, NY, USA), WESS '11, ACM, 2011, pp. 2 :1–2 :9.

[MV10] R. Maes and I. Verbauwhede, *Physically unclonable functions : A study on the state of the art and future research directions*, Towards Hardware-Intrinsic Security (A-R. Sadeghi and D. Naccache, eds.), Information Security and Cryptography, Springer Berlin Heidelberg, 2010, pp. 3–37 (English).

[MVHV12] R. Maes, A. Van Herrewege, and I. Verbauwhede, *Pufky : A fully functional puf-based cryptographic key generator*, Proceedings of the 14th International Conference on Cryptographic Hardware and Embedded Systems (Berlin, Heidelberg), CHES'12, Springer-Verlag, 2012, pp. 302–319.

[Ngo12] X. Ngo, *Implémentation et caractèrisation de la structure tero puf*, 2012, Rapport de Stage.

[NIS12] NIST, *Recommendation for the entropy sources used for random bit generation*, 2012, http ://csrc.nist.gov/publications/drafts/800-90/draft-sp800-90b.pdf.

[OHS08] E. Öztürk, G. Hammouri, and B. Sunar, *Physical unclonable function with tristate buffers.*, ISCAS, IEEE, 2008, pp. 3194–3197.

[Pap01] R. Pappu, *Physical One-Way Functions*, Ph.D. thesis, Massachusetts Institute of Technology, March 2001.

[PRTG02] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, *Physical One-Way Functions*, Science **297** (2002), no. 5589, 2026–2030, DOI : 10.1126/science.1074376.

[RSS+10] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, *Modeling attacks on physical unclonable functions*, Proceedings of the 17th ACM conference on Computer and communications security (New York, NY, USA), CCS '10, ACM, 2010, pp. 237–249.

[Sap11] S.S. Sapatnekar, *Overcoming variations in nanometer-scale technologies*, Emerging and Selected Topics in Circuits and Systems, IEEE Journal on **1** (2011), no. 1, 5–18.

[SD07] E. Suh and S. Devadas, *Physical unclonable functions for device authentication and secret key generation*, DAC, 2007, pp. 9–14.

[SHO07] Y. Su, J. Holleman, and B.P. Otis, *A digital 1.6 pj/bit stable chip-id generating circuit using process variations*, 406 –611.

[SHO08] _____, *A digital 1.6 pj/bit chip identification circuit using process variations*, Solid-State Circuits, IEEE Journal of **43** (2008), no. 1, 69 –77.

[Sim91] G.J. Simmons, *Identification of data, devices, documents and individuals*, Security Technology, 1991. Proceedings. 25th Annual 1991 IEEE International Carnahan Conference on, oct 1991, pp. 197 – 218.

[SKA+11] G. Selimis, M. Konijnenburg, M. Ashouei, J. Huisken, H. de Groot, V. van der Leest, G.-J. Schrijen, M. van Hulst, and P. Tuyls, *Evaluation of 90nm 6t-sram as physical unclonable function for secure key generation in wireless sensor nodes*, Circuits and Systems (ISCAS), 2011 IEEE International Symposium on, may 2011, pp. 567 –570.

[SSAQ02] D. Samyde, S. Skorobogatov, R. Anderson, and J-J Quisquater, *On a new way to read data from memory*, Security in Storage Workshop, 2002. Proceedings. First International IEEE, 2002, pp. 65–69.

[Tec] Verginia Tech, 2012. Research on Physical Unclonable Functions (PUFs) at SES Lab, Virginia Tech. http ://rijndael.ece.vt.edu/puf.

[VD10] M. Varchola and M. Drutarovsky, *New high entropy element for fpga based true random number generators*, 2010.

[Xil] Xilinx, *Virtex-5 libraries guide for hdl designs*, http ://www.xilinx.com/support/documentation/sw_manuals/xilinx14_4/virtex5_hdl.pdf.

[YKL+13] Y. Yao, M. Kim, J. Li, Igor L. Markov, and F. Koushanfar, *Clock-puf : Physical unclonable functions based on clock networks*, Design, Automation Test in Europe Conference Exhibition (DATE), 2013, 2013, pp. 422–427.

[YSI+11] D. Yamamoto, k. Sakiyama, M. Iwamoto, K. Ohta, T. Ochiai, M Takenaka, and K. Itoh, *Uniqueness enhancement of puf responses based on the locations of random outputting rs latches*, CHES, 2011, pp. 390–406.

Annexe A

Introduction

A.1 Contexte et Motivations

En raison de l'utilisation croissante des appareils électroniques dans tous les aspects de la vie quotidienne dans un large domaine d'applications, le besoin de sécuriser l'information a augmenté de façon exponentielle ces deux dernières décennies. Ainsi, le piratage de logiciels, la contrefaçon du matériel et la sécurité des appareils électroniques, sont devenues des problèmes prioritaires pour les industriels.

La cryptographie est une science utilisée pour remédier à ces problèmes de sécurité. En fonction du système à sécuriser et la nature de l'information secrète, une ou toutes les mesures de sécurité suivantes sont appliquées : l'authentification, l'intégrité, la confidentialité et la non-répudiation. Cependant, le niveau de sécurité est très dépendant de la clé utilisée dans le cas du chiffrement, et l'identifiant dans le cas de l'authentification.

Dans le but de voler des données privées ou de briser un protocole d'authentification, l'attaque la plus connue consiste à intercepter la clé/l'identifiant secret(e). À la fois, la production et le stockage de la clé/l'identifiant doivent être sécurisé contre tout type d'attaques. Cependant, les logiciels de génération de clés sont déterministes et donc la clé générée est prévisible. De plus, le stockage de la clé/l'identifiant dans une mémoire non volatile est vulnérable aux attaques. Par conséquent, des techniques de sécurité appropriées doivent être appliquées pour faire face à ces problèmes de sécurité.

Les fonctions non clonables physiquement (PUF) semblent être une solution alternative aux techniques cryptographiques classiques. Elles sont le sujet principal de cette thèse. Les PUF en silicium peuvent être définies comme des fonctions physiques qui fournissent une réponse unique au dispositif en se basant sur les variations intrinsèques lors du processus de fabrication du circuit. Ce dernier donne aux PUF quelques propriétés intéressantes qui en font une solution de sécurité appropriée. Nous notons trois principales propriétés d'une PUF :

- imprévisibilité : la réponse générée par la PUF varient d'une façon aléatoire d'un circuit à un autre. Mais elle est statique sur un même circuit.
- non clonabilité : la variation aléatoire du processus de fabrication rend la structure PUF très difficile à cloner.
- robustesse : la PUF doit être robuste contre les attaques physiques. Par exemple, les attaques invasives ne doivent pas être en mesure de forcer la réponse PUF, ou doivent être détecté et évitées.

Nos objectifs dans ce travail de recherche sont de concevoir et de caractériser des PUF en silicium qui répondent à ces trois propriétés. Nos priorités sont de faciliter leur mise en œuvre, les rendre portables sur différentes plateformes, et les protéger contre les attaques physiques et/ou mathématiques. En 2000, Lofstrom et al. [LDT00] ont proposé la première PUF. Depuis, au moins une nouvelle structure de PUF est proposée chaque année. Contrairement aux tests statis-

tiques proposés par le NIST [NIS12], le BSI [KS11] ou le FIPS [FIP01] utilisés pour évaluer la robustesse des structures TRNG, aucun test standard n'a été encore défini pour évaluer et comparer les performances des PUF.
Par conséquent, dans ce travail de thèse, nous proposons également des méthodes et métriques d'évaluation des PUF.

A.2 Plan de la Thèse et Contributions

Dans cette thèse, l'accent est mis sur l'étude des PUF en silicium, leur structure, propriétés et applications. Plus concrètement, les principales contributions sont les suivantes :

- Proposition de nouvelles structures de PUF faciles à intégrer qui résistent aux attaques physiques.
- Développement d'une nouvelle méthode d'évaluation des PUF à délais.
- Application de ces fonctions pour la génération de clés de chiffrement et pour l'authentification de circuits intégrés.

Cette thèse est organisée comme suit :

Le chapitre 1 présente le contexte des fonctions non clonables physiquement (PUF). Il clarifie le concept de PUF et détaille les propriétés de base qu'une PUF doit vérifier. Ce chapitre présente quelques applications utilisant des PUF et définit les différentes classifications des PUF proposées dans la littérature. Une exploration en profondeur des structures de PUF en silicium les plus connues, leurs mises en œuvre et leurs performances sont également discutées dans ce premier chapitre. En outre, un aperçu des méthodes d'évaluation est présenté.

Le chapitre 2 présente une nouvelle architecture de PUF en silicium à délais. Cette structure est appelée "loop PUF". Elle est basée sur des éléments à retard identiques et contrôlables. Ils sont connectés en série et fermés par un inverseur pour former un seul oscillateur en anneau. Ce chapitre présente les stratégies de mise en œuvre ainsi que les performances de l'"arbiter PUF" et la "loop PUF".

Nous étudions les performances de ces PUF à délais avec des implantations matérielles sur ASIC et FPGA de technologie CMOS 65 nm.

Le troisième chapitre se concentre sur la présentation d'une autre structure PUF, sa mise en œuvre sur les plates-formes FPGA ainsi que l'évaluation de ses performances. La nouvelle structure PUF est nommée "TERO PUF". Elle profite de la métastabilité oscillatoire introduite par une bascule SR. Ce chapitre détaille les étapes de mise en œuvre nécessaires à la validation de la structure TERO PUF. Il présente également les résultats d'évaluation de la performance de la structure proposée lorsqu'elle est conçue sur un FPGA ALTERA Cyclone II.

Le chapitre 4 propose une nouvelle méthode d'évaluation de la performance des PUF à retard. Elle utilise des mesures statistiques sur les éléments à retard.

Son avantage vient de sa capacité à assurer au concepteur que sa PUF présente de bonnes performances avant son intégration. Dans ce chapitre, nous détaillons notre méthode en présentant de nouvelles mesures et les résultats d'évaluation à la fois de l' "arbitre PUF" et la "loop PUF" embarquées sur différentes plates-formes.

Le sujet abordé dans le chapitre 5 est lié aux applications de la "loop PUF"proposée dans le chapitre 2. Dans ce chapitre, nous commençons par présenter nos motivations dans l'utilisation de la "loop PUF" pour l'authentification des circuits intégrés et la génération des clés cryptographiques. Puis, nous détaillons la procédure d'authentification proposée et nous montrons les résultats obtenus lors des tests sur des plates-formes ASIC. La méthode proposée est basée sur la mesure de grandeurs physiques des éléments de retard. Ces valeurs physiques sont utilisées pour authentifier les circuits car elles sont beaucoup plus précises que la réponse après quantification binaire de la "loop PUF". La méthode d'authentification proposée est en effet basée sur le coefficient de corrélation de Pearson. Ensuite, nous décrivons la procédure de génération de clés qui a été développée pour la loop PUF. Pour terminer nous discutons quelques résultats obtenus lors des tests sur les plate-formes ASIC. La méthode proposée peut être divisée en deux étapes : le profilage et la génération de clés en mode utilisateur. Afin d'améliorer la fiabilité de la clé générée, nous proposons une procédure d'analyse de fiabilité dynamique. Cependant, des fois, cela ne suffit pas pour garantir la régénération de la clé de référence. Par conséquent, nous proposons une procédure de correction qui est basée sur les codes de Hamming et l'algorithme Chase.

Enfin, le dernier chapitre fournit des remarques générales finales et met en lumière des perspectives pour de futures recherches.

Annexe B

Résumé des Chapitres de la Thèse

B.1 Les Fonctions Non Clonables Physiquement : Concept de Base

Ce chapitre présente les connaissances de base nécessaires au lecteur pour une bonne compréhension de ce document. Nous clarifions la notion de fonctions non clonables physiquement (PUF)

Nous discutons d'abord le concept de PUF, deuxièmement, nous détaillons les propriétés les plus connues que doivent posséder les PUF, ainsi que ses différentes applications. Nous définissons ensuite les différentes classifications de ces fonctions physiques proposées dans la littérature. Nous fournissons ensuite une exploration détaillée des types de PUF en silicium les plus connus, nous présentons ce qu'implique leur mise en œuvre ainsi que leur performance. Enfin, un aperçu des méthodes d'évaluation existantes est présenté à la fin de ce premier chapitre.

B.2 Loop PUF

Dans ce chapitre, nous présentons notre première contribution aux travaux de recherche sur les PUF. Nous présentons une nouvelle structure de PUF en silicium nommée "Loop PUF". Pour ce fait, nous commençons tout d'abord par détailler son architecture et ses avantages. Par la suite nous présentons les détails de son implémentation dans deux plates-formes ASIC et FPGA utilisant la même technologie CMOS 65 nm. Enfin, une analyse des performances de la structure est proposée, en effet, nous la comparons avec une autre structure de PUF en silicium connue sour le nom d'"arbiter PUF".

Cette nouvelle structure de PUF, "Loop PUF", présente deux apports majeurs :

- elle permet d'augmenter d'une manière significative la fiabilité de la réponse de la PUF.
- elle est facile à concevoir, elle n'exige aucune contrainte sévère de placement ou de routage lors de sa conception quelque soit la plate-forme FPGA ou ASIC.

Le principe de la structure proposée est d'assembler en série des chaînes à délais contrôlables. Ces chaînes sont reliées par un inverseur afin de créer un seul oscillateur en anneau. À partir des comparaisons des fréquences d'oscillation de la structure PUF en utilisant des mots de contrôle différents, on génère la réponse de la Loop PUF. Notons que la fiabilité de réponse du la "loop PUF" est renforcée en augmentant la fenêtre de mesure, ce qui équivaut à répéter plusieurs tests dans le cas de l'"arbiter PUF". L'intérêt de cette structure par rapport à celle du "RO PUF" est qu'il n'y a qu'un seul oscillateur, ce qui évite les attaques en couplage des fréquences. En outre, l'intérêt en terme de temps de développement est qu'une

chaîne de référence a pu être conçue a priori, et qu'elle peut être simplement réutilisée lors de l'intégration du "Loop PUF" dans un système électronique.

Dans ce chapitre, nous avons comparé la "loop PUF" à l'"arbiter PUF" en utilisant deux plates-formes conçues avec la technologie 65 nm : ASIC et FPGA. L'analyse des résultats est effectuée dans des conditions environnementales différentes. Les expériences nous ont permis d'étudier :

- l'impact de la technologie CMOS 65 nm sur les PUF à délais en ASIC et en FPGA .
- la comparaison entre deux structures de PUF à délais conçu avec le même type de plate-forme (ASIC).

Dans une premier temps, nous concluons que les loop et arbiter PUF présentent de meilleures performances en ASIC quand FPGA. Tout d'abord, le caractère aléatoire de la PUF arbitre est meilleure en ASIC. Deuxièmement, le caractère unique interne des deux structures est meilleure en ASIC.

En comparant les performances des deux structures étudiées en ASIC, nous concluons que l'unicité de la réponse de la "loop PUF" est meilleure que celle de l'"arbiter PUF". Indépendamment de son emplacement dans l'ASIC, la "loop PUF" présente des performances similaires en termes d'unicité de sa réponse.

B.3 TERO PUF

Dans ce chapitre, nous proposons une nouvelle structure de PUF qui exploite la métastabilité oscillatoire d'éléments couplés en croix. Cette structure est appelée "TERO PUF". Le principal avantage de cette structure est qu'elle n'est pas sensible aux phénomènes de verrouillage. En effet, contrairement à la RO-PUF, la fréquence d'oscillation n'est pas prise en compte. La PUF proposée utilise le nombre d'oscillations en tant que source d'entropie.

Nous commençons par une présentation de la structure cellulaire TERO. Nous détaillons l'étape de mise en œuvre nécessaire à la validation de la structure TERO PUF. Nous listons ensuite les problèmes qui peuvent survenir et les solutions appliquées. Par la suite, nous étudions comment cette structure peut être utilisée afin d'extraire l'entropie de la variation du processus de fabrication et de générer une réponse. Ensuite, nous évaluons les performances de la structure TERO PUF, cette structure a en effet été implantée 64 fois sur 9 FPGAs de type ALTERA Cyclone II. L'étude expérimentale montre que la structure proposée fournit des réponses avec des performances d'unicité et de stabilité intéressantes. De plus, nous avons conclu que cette même structure peut être utilisée comme un générateur de nombres aléatoires vrai (TRNG).

B.4 Méthode d'Évaluation des Performances des PUF à Délais

Dans ce chapitre, nous proposons une nouvelle méthode de caractérisation qui a pour but une meilleure évaluation des PUF à retard, dès l'étape de conception. Avec la croissance continue des architectures PUF, il devient nécessaire d'étudier leurs méthodes d'évaluation. Toutes les méthodes existantes dans la littérature sont basées sur des tests statistiques sur la réponse binaire du PUF. Dans ce chapitre, nous proposons une méthode de caractérisation qui est spécifique aux PUF à retard. Elle est basée sur l'étude des mesures statistiques des éléments à retard de la structure à étudier. L'avantage de cette méthode vient de sa capacité à permettre au concepteur de s'assurer des performances de sa structure avant son implantation. Dans ce chapitre, nous fournissons des détails au sujet de notre méthode en présentant de nouvelles mesures et les résultats de l'évaluation à la fois de l'"arbiter PUF" et de la "loop PUF" lorsqu'elles sont conçues sur différentes plates-formes.

En effet, les méthodes classiques d'étude des performances ont besoin d'un nombre considérable d'essais afin d'exécuter une méthode d'estimation de Monte-Carlo. Cependant, notre méthode nécessite moins d'essais pour évaluer la performance d'une structure PUF. Les indicateurs étudiées pour caractériser les PUF sont :

- L'aléa de la réponse de la PUF : Idéalement le nombre de 0 et de 1 dans la réponse de la PUF doivent être identiques.
- L'unicité de la réponse de la PUF : Deux PUF identiques doivent fournir deux réponses différentes à un même challenge.
- La stabilité de la réponse de la PUF : Une PUF donnée doit idéalement fournir la même réponse à même challenge quelque soit l'environnement de test.

Avec la méthode proposée dans ce chapitre, le nombre d'essais nécessaires à l'évaluation des indicateurs de performance d'une PUF est en effet linéairement croissant avec le nombre d'éléments à retard composant la PUF. De plus, lorsque nous effectuons la simulation électrique des retards des éléments de base de la structure PUF, cette méthode peut être appliquée à l'étape de conception. Un concepteur peut donc évaluer la performance de sa PUF avant sa mise en œuvre. Bien que cette méthode possède plusieurs avantages, elle a aussi deux inconvénients. Tout d'abord, la caractérisation d'une PUF ne peut être réalisée que par le concepteur ou par quelqu'un qui a accès à la structure de la PUF. Deuxièmement, cette méthode est dédiée aux PUF à délais.

Afin de comparer notre méthode avec les méthodes de caractérisation basée sur des valeurs logiques de la réponse de la PUF (par exemple, la méthode de Hori), nous avons évalué les structures loop et arbiter PUF mis en œuvre dans une la plate-forme ASIC en utilisant les deux méthodes. Les résultats montrent

que les deux méthodes présentent des résultats similaires pour l'unicité et la stabilité. Et pour le caractère aléatoire, la méthode proposée est beaucoup plus précise, car les résultats ne dépendent pas des challenges utilisés.

B.5 Loop PUF : Authentification des Circuits Intégrés et Génération de Clés Cryptographiques

Le sujet abordé dans ce chapitre est lié aux applications de la structure "loop PUF" proposée dans le Chapitre 2. Nous présentons d'abord l'interêt de la "loop PUF" dans le cadre de l'authentification des circuits et la génération de clés cryptographiques. En effet, la "loop PUF" possède des propriétés intéressantes qui pourraient contribuer à augmenter la fiabilité de ces applications.

- **Valeur de sortie précise** : La "loop PUF" fournit en sortie une valeur entière qui représente la fréquence de la boucle pendant une période fixée.
- **Grand nombre de challenges** : La fiabilité de la "loop PUF" augmente lorsque nous choisissons les meilleurs challenges. Cela nous permet d'utiliser la "loop PUF" dans des applications plus rigoureuses comme la génération de clés de chiffrement à faible coût.

Dans un premier temps, nous détaillons la méthode proposée pour l'authentification des circuits intégrés. Cette méthode est scindée en deux étapes :

1. **L'apprentissage.** Elle est réalisée une seule fois par le concepteur afin de déduire le vecteur de référence.
2. **L'authentification.** Elle est réalisée à chaque fois qu'une authentification du circuit est demandée.

La méthode que nous proposons est basée sur le calcul du coefficient de Pearson comme métrique d'authentification. Grâce à la précision de la sortie de la "loop PUF" et la méthode que nous proposons, nous sommes en mesure de distinguer un exemplaire d'un autre quelle que soit sa position dans l'ASIC, sans post-traitement particulier et avec une marge d'erreur confortable. La procédure d'authentification est rapide, le temps d'authentification est linéairement dépendant du nombre d'élément à retard de la Loop PUF. Dans notre cas, en utilisant une structure de PUF de 64 éléments à retard, nous avons besoin de 65 mesures pour authentifier un PUF. Avec une fréquence d'horloge de système de 100Mhz , nous sommes en mesure d'effectuer toutes les mesures nécessaires en $1,08ms = 65 * T_{puf}$ ce qui est très faible.

Ensuite, nous décrivons la procédure configurable de génération de clés qui a été développée en utilisant la Loop PUF. La méthode proposée est scindée en deux étapes :

1. **Le profilage** : Cette étape peut être considérée comme une étape d'apprentissage de la PUF et de son comportement dans le circuit. Elle consiste à identifier la clé de référence avant la mise en vente du système.

2. **La génération de clé** : Cette étape est rélisée par l'utilisateur de la PUF. Le but de cette étape est de permettre à l'utilisateur de regénérer une clé identique à celle référence.

La méthode que nous proposons tire profit des techniques suivantes :
- Élargissement de la fenêtre de mesure
- Sélection des meilleurs paires de challenges.
- Augmentation du nombre d'essais.
- Utilisation d'une procédure de correction de clés.
- Suppression des bits non fiables.

Selon les contraintes fixées (consommation en temps, la complexité du correcteur ou la longueur de la clé désirée) et le taux d'échec maximum toléré (BER), nous pouvons configurer la procédure de génération de clés. L'analyse des performances de la procédure proposée pour la génération de clés de chiffrement est effectuée sur deux plates-formes ASIC. Nous avons conclu que, en utilisant la structure de la "loop PUF" et la procédure de génération de clés proposée, nous sommes en mesure de générer une clé cryptographique fiable de 61 bits avec un taux d'échec $\mathsf{BER} < 10^{-9}$ en $\mathsf{10ms}$.

Annexe C

Conclusions et Perspectives

C.1 Conclusions

Les fonctions non clonables physiquement ou PUF sont des primitives de sécurité physique émergentes. Elles sont capables de protéger les dispositifs contre les problèmes de contrefaçon, de protéger les processus de génération et de stockage des clés de chiffrement contre les attaques physiques et mathématiques, etc. Dans ces travaux de thèse, nous nous sommes concentrés sur les PUF en silicium, leurs applications, leurs structures, ainsi que les méthodes qui permettent de les caractériser.

Dans le but de faciliter la mise en œuvre et l'amélioration des PUF existantes, nous avons proposé deux structures de PUF en silicium. Notre première contribution a été consacrée à la description et à l'évaluation d'une structure de PUF à délais appelée "Loop PUF". Nous avons montré que l'architecture proposée est aisée à concevoir quelque soit la plate-forme choisie. En outre, et en raison de sa structure non différentielle, elle permet de remédier aux difficultés de placement et de routage. Nous avons également montré que la fiabilité de la structure proposée peut être améliorée au cours de l'acquisition des mesures des fréquences d'oscillation de la structure sans avoir besoin de procédés de post-traitement. Ceci est possible en augmentant la taille de la fenêtre de mesure ainsi qu'en augmentant le nombre de tests effectués. Ensuite, la structure proposée a été évaluée sur deux plates-formes FPGA et ASIC utilisant la même technologie CMOS 65 nm. Des expériences pratiques ont montré que la structure proposée présente des performances intéressantes. La seconde structure PUF proposée est une PUF basée sur des cellules temporairement oscillantes. Elle est appelée "TERO PUF" . Nous avons montré que le principal avantage de cette structure est qu'elle n'est, théoriquement, pas sensible au phénomène de verrouillage. Le processus de mise en œuvre ainsi que les résultats expérimentaux de cette structure sur une plate-forme FPGA ALTERA sont détaillés dans ce manuscrit. Nous avons conclu que la "TERO PUF" propose des performances d'unicité et de stabilité intéressantes.

En troisième contribution, nous avons proposé une nouvelle méthode pour caractériser les performances des PUF à délais. Elle tire profit des mesures des grandeurs physiques. L'intérêt principal de cette méthode est que, contrairement aux méthodes classiques, elle ne nécessite pas un grand nombre d'essais pour évaluer la performance de la PUF. En outre, elle peut être appliquée à l'étape de conception. Nous avons identifié les propriétés les plus importantes auxquelles une PUF doit répondre et nous avons proposé par la suite les métriques correspondantes. Nous avons également validé les mesures proposées par une comparaison entre les résultats de la caractérisation de deux structures de PUF à retard en utilisant notre méthode et une déjà existante basées sur des tests statistiques sur la sortie logique de la fonction.

Ensuite, nous avons proposé deux applications de sécurité utilisant la structure Loop PUF. Nous avons d'abord étudié l'utilisation de la réponse de la PUF

pour authentifier un circuit intégré (IC) .

Nous avons proposé une méthode basée sur la mesure des grandeurs physiques des éléments à délais de la “loop PUF” et sur le coefficient de Pearson comme une métrique d’authentification. Sur cette base, la performance de cette méthode d’authentification est évaluée. Deuxièmement, nous avons proposé une nouvelle méthode de génération de clé cryptographique basée sur l’utilisation de la Loop PUF. Nous avons commencé par l’étude des techniques existantes pour améliorer la fiabilité des données bruitées. Ensuite , nous avons proposé une méthode qui tire profit des avantages de la Loop PUF. Les résultats expérimentaux ont montré l’efficacité de cette méthode de génération de clé basée sur la Loop PUF.

C.2 Recherches Futures

Plusieurs domaines de recherche intéressants découlent de ce travail. Dans ce qui suit, nous soulignons certaines de ces directions de recherche.

Une perspective immédiate de notre première contribution réside dans l’analyse de la robustesse contre la modélisation mathématique et les attaques physiques de la structure de la “loop PUF” que nous avons proposée. Il est ensuite intéressant de renforcer notre structure en proposant une solution de sécurité à bas coût comme une contre-mesure contre les attaques possibles . Une perspective utile d’un point de vue pratique serait d’utiliser une approche algorithmique pour le processus de génération des challenges de la structure “loop PUF” fondée sur notre recherche théorique [CDG+13].

Les recherches futures sur les travaux autour de la structure “TERO PUF” sont l’évaluation de sa robustesse contre les attaques électro- magnétiques afin de valider nos hypothèses théoriques. Une autre perspective pour cette contribution est d’optimiser la structure de la “TERO PUF” afin d’améliorer sa complexité. Il serait également utile d’étudier l’utilisation des bits de poids faible du compteur de la boucle TERO pour générer des nombres aléatoires. Ainsi, la cellule “TERO PUF” proposée servira à la fois comme un TRNG et comme une PUF.

Une perspective immédiate de la méthode de caractérisation proposée est de valider qu’elle nous permet d’obtenir les mêmes résultats d’évaluation lorsqu’on utilise un modèle de simulation qu’en utilisant une réalisation physique d’une structure donnée de PUF à délais. Une extension importante également, d’un point de vue théorique, est d’étendre les mesures proposées pour d’autres constructions PUF, par exemple les RO PUF.